COMPUTER TYPESETTING

Experiments and Prospects

COMPUTER TYPESETTING

Experiments and Prospects

MICHAEL P. BARNETT

THE M.I.T. PRESS
MASSACHUSETTS INSTITUTE OF TECHNOLOGY
CAMBRIDGE, MASSACHUSETTS, AND LONDON, ENGLAND

Copyright © 1965 by
The Massachusetts Institute of Technology
All Rights Reserved. This book may not be
reproduced, in whole or in part, in any form
(except by reviewers for the public press),
without written permission from the publishers.

Library of Congress Catalog Card Number: 65–27789
Printed in the United States of America

To Barbara

Preface

In February 1961 the author encountered the tape-operated Photon 200 phototypesetting machines at the Machine Composition Company in Boston, largely by accident. The possibility of producing operating tapes for these machines from the output of a digital computer seemed a rather interesting matter to pursue, and a computer program was coded in the summer of 1961 that had as its peak of achievement the production of a tape that set the "Tail" from *Alice in Wonderland* (see p. viii). The program also was used to produce a phototypeset press release (see p. xii) and a progress report. It did nothing else that was particularly useful, however, and the work on typesetting in the author's laboratory sagged for a year. Following the award of a research grant to the project in 1962, further staff joined. A Photon 560 machine was installed, and the coding of a system of computer programs that was called the PC6 system was completed. This was used in late 1963 and in early 1964 to set several hundred pages of material for a variety of reports, papers, pamphlets, and other publications that were of interest to the author and to his colleagues, prior to his taking up an appointment at the University of London and the transfer of the equipment to the University of Pittsburgh.

The interaction of computers and typesetting devices has attracted a great deal of attention throughout the computing and typesetting industries in recent months. The number of projects that are concerned with such work may well have reached the hundred mark and

```
[indn77dl21s24st1,,36cnxs1]
EXCERPT FROM ALICE IN WONDERLAND
[nl1sl8]
December 6, 1961
[sp4st2,10,36st3,11,36st4,12,36st5,13,36st6,14,36s
st8,16,36st9,17,36st10,18,36ls14dl1xs2rl]
[sc19sc19]Fury said to
[xs3]a mouse, That
[nlxs6]he met
[xs7]in the
[xs8]house,
[xs9][sc19]Let us
[xs7]both go
[xs6ls12]to law:
[xs5dl9]I[dl1] will
[xs3]prosecute
[xs2dl9]you.
[nlxs3dl1] Come, I'll
[nlxs4]take no
[nlxs5]denial:
[nlxs7]We must
[nlxs8ls11]have a
[nlxs9]trial[sc47]
[nlxs10] For
[xs7]really
[xs6]this
[nl] morning
[nlxs8ls10] I've
[xs6]nothing
[xs5]to do.'
[xs4]Said the
[xs2]mouse to
[xs1] the cur,
[nlxs3][sc19]Such a
[nlxs4ls9]trial,
[xs3]dear sir,
[xs2] With no
[xs2]jury or
[xs1ls8]judge,
[nlxs2]would be
```

"Fury said to
a mouse, That
he met
in the
house,
'Let us
both go
to law:
I will
prosecute
you.
Come, I'll
take no
denial:
We must
have a
trial;
For
really
this
morning
I've
nothing
to do.'
Said the
mouse to
the cur,
'Such a
trial,
dear sir,'
With no
jury or
judge,
would be
wasting
our breath.
'I'll be
judge,
I'll be
jury,'
Said
cunning
old Fury:
'I'll try
the whole
cause.

"The Tail:" Phototypeset output and portion of Flexowriter input.

computers are being installed in typesetting environments for production work in the newspaper and the book publishing industries in the United States and elsewhere.

The PC6 system that is reported in the first half of this monograph did not constitute a production system that was ready for industrial use. This is reflected by the fact that the system has not been used to set this account of it. It was planned as an experimental system, and it was used for a number of typesetting experiments. The case for reporting such experiments now is by no means dismissed by the industrial applications of computer-aided typesetting that are going into operation. The work that has been done to date in industry, government agencies, and universities just scratches at the surface of the field's potentialities. The provisions of the programs that are described in this monograph are patently weak, and they are subject to hosts of improvements that in many cases could have been made individually with relatively minor effort. The material is being reported in the present form and at the present juncture, not in ignorance of this fact, but in part because of it. Every improvement is likely to open up the possibility of further improvements. There is no end to the over-all situations that give rise to typesetting jobs, or to the details of such situations that could be accommodated to advantage in the programs that mechanize the typesetting process. There is no end to the format characteristics that such programs should enable the user to control, or to the ways in which these might be specified conveniently. The author is aware that many of the details of the work that is reported in the first half of this book relate to features of the Photon 560 machine which are shared by only a minute fraction of the typesetting machines that are in use today. There are further details that result from his lack of experience of the printing industry. In particular, some of the more tortuous features of the programs were designed for typographic horrors that may never arise in practice. The material in this work that may be of some general interest could not be presented, however, without mentioning these more parochial details. It is hoped that the PC6 programs achieved some usefulness in three respects. First, the flexibility of format control which the programs provided has allowed experiments to be performed that may lead to the specification of the outward characteristics of systems which will be of more practical value. Second, the internal organization of the PC6 system may provide ideas that will be of use in the design of the internal structure of further programs which will allow more extensive control of the format characteristics of the typeset output and further variety in the forms of input that may be employed. Third, the actual use of the programs has expedited the dissemination in printed form of some scientific and other work in a way that may have been of use to its authors and to its readers.

This monograph was intended originally as a summary of the computer-aided typesetting work of the author and his colleagues at M.I.T. Several further chapters were written, however, in an attempt to put this work into perspective by surveying the more general issues of computer-aided typesetting. These further chapters tend to raise questions rather than to answer them, but it is the printing-publishing industry which in large measure must provide the answers.

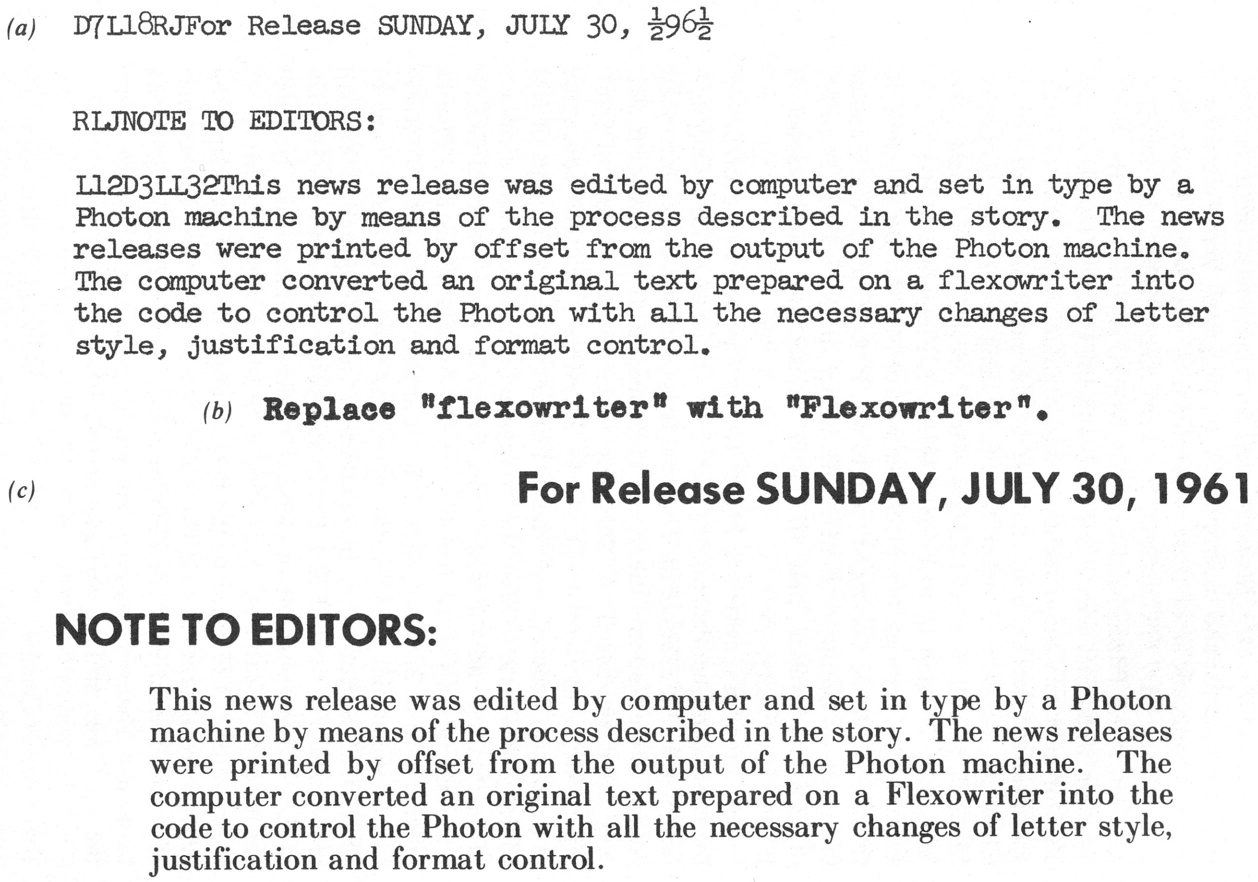
An effort has been made throughout this monograph to convey to the readers certain general ideas concerning computing. First and foremost, there is the need for members of an industry that intends to use computers to specify the outward characteristics of the programs that should be written to meet their requirements. Of particular importance too is the recognition that computer programs and applications evolve, and that good computer usage provides flexibility rather than imposing restrictive standardization. The mechanics of achieving this flexibility by fragmentation of an elaborate process for which a program is written is perhaps the main technical issue which this monograph tries to present. The separation of the effects of a program from the way that the user specifies these effects is an associated major motif that runs through much of the discussion of more specialized topics in the chapters that follow.

The work that is reported in the first half of this monograph was conducted in the M.I.T. Cooperative Computing Laboratory (C.C.L.), of which the author was Director. The C.C.L. housed an IBM 709 computer that provided an interdepartmental computing facility which was used by several hundred members of M.I.T. for a very wide range of computing activities. The typesetting project was just one of these. The convenience of conducting the typesetting research within the C.C.L. determined the choice of computer. The specification and acquisition of the Photon 560 phototypesetting machine was a logical sequel to the initial work that used the Photon 200 machines at the Machine Composition Company in Boston, though the original encounter with these, within the totality of tape-operated typesetting equipment, was largely fortuitous.

Several members of the C.C.L. staff played major parts in the development and applications of the typesetting programs that are reported in the first half of this book. In particular, Dr. D. A. Luce was responsible for the detailed supervision of much of the work during the year 1962–3. A considerable amount of the programming work was done by Mr. E. J. Desautels, Mr. K. L. Kelley, Mr. C. R. Morgan, and Mr. D. J. Moss. Further contributions to the basic typesetting programs, and to supplementary programs that were used in various applications of these, were made by Mr. M. J. Bailey, Mr. J. E. Clancy, Mr. T. Gebsattel, Mr. J. M. Gerard, Miss M. A. Hartnett, Dr. R. M. Pitzer, Miss J. Tonso, Mr. A. B. Otis, and Mr. H. D. Wactlar, when they were associated with the C.C.L. Thanks are due also for help in various technical and organizational matters in the specification, development, and use of the programs to Mr. W. J. Clements, Mr. D. C. Crocker, Mrs. E. B. Langford, Dr. W. N. Locke, Mr. J. I. Mattill, Mr. R. M. Ross, Mr. P. Scott, Mr. J. Sheetz, and Mrs. B. P. N. Shih of M.I.T., to Mr. J. J. Fitzgerald of the M.I.T. Lincoln Laboratories, Mr. N. Sutherland of MITRE Corporation, to the late Mr. K. A. Loring, Sr., and to Mr. P. R. Kelly of the Machine Composition Company, to Mr. F. D. Miller and Mrs. M. P. Proshkin of IBM Corporation, and Mr. D. D. Friesen and his colleagues at Boston University.

The author is indebted to Mr. J. Kuney and Mr. B. Lazorchak of the American Chemical Society, Mr. E. F. Coleman of Eltra Corporation, Mr. E. J. Forbes of the National Bureau of Standards, Mr. Scott Adams of the National Library of Medicine, to Mr. J. Ashworth, Mr. J. A. Boyd, and Mr. T. A. Skingsley of the London College of Printing, to Mr. L. E. Kenyon, Mr. C. W. D. Alister, and their colleagues at the British Federation of Master Printers, to Mr. A. Phillips of Her Majesty's Stationery Office, and to several of the officers and members of the Printing and Kindred Trades Federation and National Graphical Association for the benefit of much helpful discussion. Thanks are due to Mrs. B. Salant, Mrs. J. B. Fournier, and Mrs. E. M. Ritter for the secretarial work in the typesetting project at M.I.T. and to Miss M. C. Stirton in the preparation of the monograph at London University. The work at M.I.T. was supported by a grant GM10–430 of the National Institute of General Medical Science of the National Institutes of Health from May 1962 until March 1964, and Mr. L. P. Hellman of N.I.H. gave to the project much help and guidance on general procedural matters. The work prior to May 1962 formed part of a more extensive program of research of the M.I.T. Solid State and Molecular Theory Group that was supported by the Army, the Navy, the Air Force, and the National Science Foundation.

University of London MICHAEL P. BARNETT
January, 1965



(a)

```
D7L18RJFor Release SUNDAY, JULY 30, ½96½

RLJNOTE TO EDITORS:

L12D3LL32This news release was edited by computer and set in type by a
Photon machine by means of the process described in the story.  The news
releases were printed by offset from the output of the Photon machine.
The computer converted an original text prepared on a flexowriter into
the code to control the Photon with all the necessary changes of letter
style, justification and format control.
```

(b) **Replace "flexowriter" with "Flexowriter".**

(c) **For Release SUNDAY, JULY 30, 1961**

NOTE TO EDITORS:

This news release was edited by computer and set in type by a Photon machine by means of the process described in the story. The news releases were printed by offset from the output of the Photon machine. The computer converted an original text prepared on a Flexowriter into the code to control the Photon with all the necessary changes of letter style, justification and format control.

The press release: (a) portion of Flexowriter input, (b) portion of editing corrections, (c) portion of typeset output.

Contents

1. Introduction 1

1.1 Computer-Aided Typesetting 1
1.2 C.C.L. Equipment 2
1.3 Internally Coded Input 3
1.4 Externally Coded Input 3
1.5 Some Further Applications 4
1.6 The ρ-Code and Preprocessors 4
1.7 Typographic Control Elements 5
1.8 Further Work 6
1.9 C.C.L. Typesetting Applications 6

2. Equipment 8

2.1 Choice of Equipment 8
2.2 The SPD Flexowriter 9
2.3 Processing Paper Tape for Input 9
2.4 The IBM 709 Computer 11
2.5 Preparing Paper Tape from Computer Output 11
2.6 The Photon 560 Machine 12
2.7 The Flash Lamp and the Character Disc 13
2.8 The Lens Turret 15
2.9 Character Alignment 16
2.10 Type Size 16

2.11 The Reflecting Prism 17
2.12 The Film Magazine 19
2.13 Developing Output 20
2.14 Pi-Mat 22
2.15 The Operating Tape 22
2.16 The Control Codes 23

3. Preparing TYPRINT Source Material 25
3.1 The TYPRINT Option 25
3.2 Styles of Type 26
3.3 Sizes of Type 28
3.4 Setting a Single Paragraph 30
3.5 Setting Several Paragraphs 33
3.6 Changing the Line Length and Starting a New Line 35
3.7 Changing the Style Within a Text 37
3.8 Changing the Spacing Between Lines 39
3.9 Changing Margins and Vertical Alignments 47
3.10 Changing Page Length and Page Numbering 50
3.11 Setting Special Characters 52
3.12 Horizontal Spacing Control 54
3.13 Tabular Material 55
3.14 Composite Codes (Macros) 64
3.15 Processing TYPRINT Input 64
3.16 Computer Editing 65
3.17 Run Processing 66
3.18 Film Preparation 67

4. TYPRINT Operating Experience 73
4.1 Introduction 73
4.2 Annotating Experience 73
4.3 Intellectual Convenience of Control Codes 75
4.4 Mechanical Convenience of Control Codes 76
4.5 Proofreading and Correcting TYPRINT Input 77
4.6 Editing Programs 77
4.7 Causes of Run Failures 79
4.8 Error Location Procedures 79
4.9 Improved Input Error Location 80
4.10 Input Error Bypassing Procedures 81
4.11 Control Unit Modifications 82

5. Preparing and Processing TABPRINT Input 84

5.1 Punched-Card Tables 84
5.2 A Simple Example 86
5.3 Operating Experience with TABPRINT 94

6. Some Further C.C.L. Applications 100

6.1 BCDPRINT, Catalogs, and Built-up Formulas 100
6.2 The BCDPRINT Option 101
6.3 The *e* Coefficient Table 103
6.4 The Newcomb Operator Table 109
6.5 *The Union List* 114
6.6 The *Current Serials and Journals* Catalog 125
6.7 The "Picture Book" Application 128
6.8 Setting Built-up Formulas 135

7. Programming Considerations 140

7.1 Introduction 140
7.2 Input, Storage, and Output 141
7.3 Compilers 143
7.4 Preprocessors 145
7.5 Mechanized Syntactic Analysis 147
7.6 Syntactic Analysis and Stylized Instructions 151
7.7 The ρ-ω Translator 153
7.8 Further Typographic Functions 157
7.9 External Control of Cyclic Format 162
7.10 A Procedural Language 163

8. Other Computer-Aided Typesetting Projects 165

8.1 Introduction 165
8.2 The Russian Translation Project 166
8.3 Table Setting at the NBS 172
8.4 The Newcastle Project 174
8.5 Commercial Text Setting 176
8.6 Man-Machine Interaction 179

8.7 The Bibliographic Reference Literature 180
8.8 Electron-Tube Devices 185
8.9 Other Projects 187

9. The Economics of Computer-Aided Typesetting 189
9.1 Introduction 189
9.2 Computer-Aided Typesetting Processes 189
9.3 Assessment of Benefits 193
9.4 Programming Costs 196
9.5 Equipment Costs 198
9.6 Over-all Production Speeds 199
9.7 Keyboarding 200
9.8 Typesetting Machines 206

10. Administrative and Training Problems 208
10.1 Introduction 208
10.2 New Skills and Training 208
10.3 A Prototype Syllabus 209
10.4 Employment Effects of Changed Keyboard Skills 214
10.5 Programming Tactics 218
10.6 Academic Facilities for Technological Research 222

11. Conclusion 227
11.1 The C.C.L. Experiment 227
11.2 Future Research 228
11.3 The Computer Culture and CATP's 229
11.4 Associated Activities 231
11.5 CATP's and Information-Generating Processes 232
11.6 CATP's and the Transformation of Information 232
11.7 CATP's and Information Retrieval 233
11.8 The Future 234

References 237
Index 241

COMPUTER TYPESETTING

Experiments and Prospects

I. Introduction

1.1 Computer-Aided Typesetting

Many publishing, computing, and other organizations are giving considerable attention to processes in which a computer records information on a punched paper tape, or on a magnetic tape, that is then used to operate a typesetting machine. The term "computer-aided typesetting process," or CATP for short, is used by the author for any such process in which paper or magnetic tape provides an interface between the operation of computers and of typesetting machines. The provision of this interface can have several advantages. It enables tape-operated typesetting machines to set computer output without the intervention of keyboard operators and the inevitable human errors. Computers can be used to perform sorting, updating, and other clerical processes in the preparation of typeset material of many forms. Also, computers can simplify the keyboard work when type is set from manuscript, by introducing details of typography and style to which the keyboard operators need not attend so fully as when setting type by the conventional methods. The typeset material that is produced as the end product of a computer-aided typesetting process is called the "typeset output" of the process.

The first part of the present monograph is concerned with the PC6 system of programs, which was coded in the author's laboratory to produce operating tapes for the Photon 560 phototypesetting machine on the IBM 709/90 computer. The equipment that was used is described in Chapter 2, and the PC6 programs are discussed in Chapters 3 to 7. Work has been reported by other groups that relates to several models of computers and to several forms of hot metal, photographic, and electron-beam typesetting equipment. Some of this work is described in Chapter 8, and a few more general aspects of computer-aided typesetting are discussed in Chapters 9 to 11.

1.2 C.C.L. Equipment

Text material was keyboarded in the Cooperative Computer Laboratory (C.C.L.) on a Friden Flexowriter. This has a typewriter keyboard, and it produces typed copy that has the appearance of typewritten material. The type is in a single style and size, and the lines are not justified. Whenever a key is struck to type a character on the paper in the platen, one or more holes are punched in a reel of paper tape that feeds through a punch unit, which forms part of the machine. These holes represent the character in a simple 8-bit code. Further codes are punched when the carriage-return, tabulate, and other nonprinting keys on the Flexowriter are struck. The tape advances automatically whenever a row of holes is punched. The Flexowriter is also equipped with a reading unit, through which a previously punched tape can be fed to retype the text that was punched in it; at the same time, a further tape is punched that is a complete or partial copy of the tape that is passing through the machine's reading unit.

The Photon 560, like other filmsetting machines, forms a photographic image of material that is to be printed, on a roll of continuous film. The film may be up to 100 feet long and up to 10 inches wide. In the Photon machine, the images of the successive characters are projected onto the film one by one, by the selective illumination of individual members of a set of photographic negatives of 1440 different characters. The Photon 560 machine is controlled by punched paper tape. Characters, styles, and sizes of type are selected, and horizontal and vertical spacing between successive characters and lines of print are controlled by means of 8-bit codes in the paper tape. The tapes for individual typesetting jobs on the Photon 560 are produced by digital computers which are used, for example, to translate from the codes of the Flexowriter into the codes of the 560 machine. The author employed a 32K, 12-tape IBM 709 computer because this was available in his laboratory at M.I.T. The Photon machine and the other items of equipment that were used are discussed further in Chapter 2.

1.3 Internally Coded Input

The PC6 system of programs enabled the IBM 709/90 computers to produce control tapes for the Photon 560 machine from several forms of source material or "input." Different options of the program were used for these. The TYPRINT option dealt with source material that had been typed on a Flexowriter. The format and the typographic style of the phototypeset end product were specified by "symbolic control codes" that were typed in the source material between square brackets at the beginning and wherever the typography had to change. These symbolic control codes consisted of letters that formed simple abbreviations, and numbers that specified dimensions. For example, the code LN400 specified an output line length of 400 points, and the code VP10 specified blank space in the output that was 10 points deep. A point is $\frac{1}{72}$ inch in English measure. Material that contains explicit control codes in the manner of TYPRINT source material is said to be internally coded.

The Flexowriter input to the TYPRINT option was treated as a continuous stream of information. It was broken into lines by the program, by reference to the page or column width and the type size that had been specified, and to particular codes that explicitly requested the beginning of a new line, paragraph, or page.

No knowledge of computer programming was needed to select the symbolic control codes to be inserted in a manuscript before it was typed on a Flexowriter for processing by the TYPRINT option. Editorial staff in several organizations found it easy to annotate manuscripts in this manner, and several jobs were run with material that was prepared by personnel who could not program computers. The preparation of TYPRINT input and its processing insofar as this affected the user are described in detail in Chapter 3. Operating experience with this form of input is discussed in Chapter 4.

1.4 Externally Coded Input

For certain types of application, it is irksome, and in extreme cases it is completely impractical, for technical or clerical personnel to prepare input that is fully coded internally. The number of characters in the control codes may exceed the number of characters that are to be typeset in tables, bibliographies, dictionaries, and other forms of listing that require very frequent changes of type style, spacing, and indentation. To deal with such cases it is expedient to punch input that is not internally coded, and to provide criteria that allow the requisite control codes to be introduced mechanically by the computer wherever they are needed. Special conventions can be used in such cases to delimit the input fields that are to be set in different ways. In the case that the input consists of tables that have been punched on cards or written on magnetic tape in a fixed format, it is a very simple matter for the computer to insert control codes in an appropriate manner. The TABPRINT option of the PC6 system dealt with tables in just this way. Headings, column widths, and the table arrangement were all specified in a "control deck" of cards that the computer read, before the table was provided. This TABPRINT option is described in Chapter 5. Input material such as this, which does not contain the explicit control codes at the points where they must take effect, is called externally coded input.

1.5 Some Further Applications

As stated above, one of the major benefits of computer-aided typesetting is its removal of the need for keyboard work when type is set from the results of computer operations. Results that are to be "printed" in the conventional computing sense are recorded, by a majority of computers, in "binary coded decimal" form on magnetic tape. This same form of coding is used to write information mechanically on magnetic tape from cards that have been punched in the Hollerith code. The BCDPRINT option of PC6 system dealt with input that was punched in Hollerith code on cards or written in binary coded decimal form on magnetic tape, and which was internally coded in a manner very similar to that of TYPRINT input. This option is described in Chapter 6, which also discusses two applications of it and two library jobs that were set by further variations of the PC6 system of programs.

1.6 The ρ-Code and Preprocessors

The PC6 programs dealt with TYPRINT, TABPRINT, BCDPRINT, and other styles of input by first expressing the material that was to be typeset in a certain standard code, called the ρ-code, and then translating this into the code of the Photon machine, called the ω-code. The ρ-to-ω translation process was performed by the same program in each case, but the initial production of the ρ-code was performed by a different program, called a preprocessor, for each of the forms of input.

Material that was expressed in the ρ-code was represented by a string of integers. Each character that was to be typeset was represented by an integer in the range 1 to 90. Each input space was represented by zero. Each format specification was represented by a negative integer that occurred at the point where it had to take effect. The ρ-code representation of a text was thus analogous to the TYPRINT representation, with negative integers standing in place of symbolic control codes and integers in the range 1 to 90 standing in place of the characters that were to be printed. The term ρ-string is used for the ρ-code representation of a text.

The conversions of TYPRINT, TABPRINT, and BCDPRINT input to the corresponding ρ-code representation were performed by preprocessors that formed part of the PC6 system of programs. Two other preprocessors were used for the specialized library applications that are described in Chapter 6, and many further forms of input could be converted into ρ-code by appropriate preprocessors. The advantage of this intermediate use of the ρ-code was that it allowed the relatively elaborate program which translated the ρ-code into the ω-code of the Photon machine to be retained for numerous forms of input. The construction of the individual preprocessors that formed the ρ-code in these cases did not in general require any knowledge or alteration of the ρ-ω translator and in some cases was relatively simple. Chapter 7 discusses some general issues of programming and comments briefly on the design of the preprocessors that dealt with various forms of input in the PC6 system.

1.7 Typographic Control Elements

Although the different options of the PC6 system dealt with input that was presented to the computer in different ways, the details of format and typography of the phototypeset end product that could be specified were the same, regardless of whether the input was coded internally as in TYPRINT, or externally as in TABPRINT. The typographic flexibility that was possible with the PC6 system was limited to formats that could be defined in terms of the individual properties to which the different control codes related. These properties, and to a lesser extent their numerical representations in the ρ-code, characterized the PC6 system.

The bulk of the PC6 system resided in a collection of about 60 subroutines which formed the ρ-ω translator. The subroutine organization of this translator was intended to facilitate the extension of the range of typographic details that could be specified by new control codes. A few comments on the design of the ρ-ω translator are included in Chapter 7, together with some suggestions concerning further typographic functions for future typesetting programs.

1.8 Further Work

Chapter 8 gives brief descriptions of some computer typesetting projects that have been conducted or are now in progress in other laboratories. Chapter 9 enumerates the range of typesetting situations in which it may be possible to use the computer to advantage and discusses some of the factors that bear on the economics of such applications. Chapter 10 considers some of the educational needs and administrative problems that arise in computer-aided typesetting work. Chapter 11 relates the field of computer-aided typesetting to some more general current trends in information science.

1.9 C.C.L. Typesetting Applications

As stated in the Preface, the PC6 programs were intended to provide a vehicle for gaining experience and insight into computer-aided typesetting practicalities by the use of these programs in actual jobs and by discussion of the usage of these programs with members of the graphic arts industry. The programs were used in the C.C.L. to set several reports for actual distribution, and the programs were used further to set a variety of shorter texts for illustrative, test, and instructional purposes. This work was done by members of the C.C.L. and by others. The programs' usage, potentialities, and limitations have also been reported in a variety of contexts, and the experience of these activities has suggested a number of matters that are discussed in the chapters of this book that follow.

The main jobs that were set for distribution by the PC6 programs were the following:

1. "Computers and Typesetting," press release issued by the M.I.T. Department of Public Relations, July 31, 1961 (4 pages). See the Preface of this monograph, p. xii.
2. "Computer Controlled Typesetting," M. P. Barnett and K. L. Kelley, *Quarterly Progress Report No. 41*, Solid State and Molecular Theory Group, M.I.T., 1961, pp. 92–94 (3 pages).
3. "Integrals Over Atomic Orbitals," Appendix II in the unpublished Ph.D. dissertation of R. M. Pitzer, Harvard University Depart-

ment of Chemistry, 1963 (25 pages). See Section 5.2 and Examples 3.26 and 5.2 of this monograph.

4. *International Union List of Communist Chinese Serials*, compiled by B. P. N. Shih and R. L. Snyder, M.I.T. Libraries, August, 1963 (121 pages). See Section 6.5 and Examples 3.28, 6.11, and 6.12 of this monograph.
5. Tables 1 and 2 in *Surface Harmonic Expansions of Products of Cartesian Coordinates*, A. B. Otis, Jr. and M. P. Barnett, Technical Note No. 26, Cooperative Computing Laboratory, M.I.T., 1963 (12 pages). See Examples 5.5 and 5.6 of this monograph.
6. Table of *e* coefficients in *Computer Generated Formulae for the* e *Coefficients of Theoretical Chemistry*, H. D. Wactlar and M. P. Barnett, Technical Note No. 31, Cooperative Computing Laboratory, M.I.T., 1963 (21 pages). See Section 6.3 and Examples 6.2 to 6.6 of this monograph.
7. Sections 4.1 through 4.5 of *Construction of Newcomb Operators on a Digital Computer*, I. G. Izsak, J. M. Gerard, R. Efimba, and M. P. Barnett, Special Report No. 140, Smithsonian Institution Astrophysical Observatory, 1964 (50 pages). See Section 6.4 and Examples 6.7 to 6.10 of this monograph.
8. Some notes prepared by the Committee on Academic Performance for the benefit of students on academic probation, M.I.T., January, 1964 (12 pages). See Example 3.30 of this monograph.
9. *Catalog of Lunar Craters*, G. S. Hawkins et al., Astronomical Contributions of the Boston University Observatory, Series II, Vol. V, No. 36 (May, 1964), (109 pages). See Examples 3.27 and 5.7 of this monograph.

The TYPRINT option was used to set item 8, a few pages of 3, 4, and 9, a short report for restricted use on certain computer programs, numerous examples for the TYPRINT manual that are reproduced in Chapter 3, and practice material by several people who were experimenting with the use of the programs. The TABPRINT option was used to set the bulk of items 3 and 9, all of 5, some examples for the TABPRINT manual that are reproduced in Chapter 5, and a 30-page table of fiscal data for restricted use at M.I.T. The BCDPRINT option was used to set 6 and 7 from computer-generated source material. Special preprocessors were used for 4 and for another library job that has not yet been run in full. The original phototypesetting program that preceded the PC6 system of programs was used to set 1 and 2.

2. Equipment

2.1 Choice of Equipment

The equipment that was employed in the author's typesetting project was determined largely by the project's environment. The Cooperative Computing Laboratory was established in 1961 as a computing facility for use by members of M.I.T. in their academic work. The C.C.L. contained an IBM 709 computer, pre-1401 off-line magnetic tape equipment, and conventional unit record equipment. A model SPD Friden Flexowriter was installed for general C.C.L. use, since similar machines were employed fairly widely in other laboratories and departments at M.I.T. The Photon 560 machine was provided by Photon Inc. in response to a request for low-cost equipment that could be operated by paper tape in the manner of the nonstandard Photon 200 machines at Machine Composition Company in Boston. It transpired that there were minor differences in coding for the Photon 560 in the C.C.L. and the Photon 200 machines at Machine Composition, but these differences affected only a very small portion of the typesetting programs. It seems that for industrial operations there is a strong advantage to a tape-operated typesetting machine for which tapes can be produced both by computer and directly from a keyboard perforator as occasion demands. The Photon 560 machine does not have this advantage. Very few Photon 560 machines have been built—the C.C.L. obtained the first—but programs could be written for other tape-operated typesetting machines to perform processes that are similar to those which are described in this book.

2.2 The SPD Flexowriter

The C.C.L. used a standard model Flexowriter[1] (see Figure 2.1) to which a small number of minor additions had been made. In a Flexowriter each hammer carries two characters. Which of these is typed depends on whether the lower- or upper-case shift key has been pressed most recently. Further keys activate ten machine functions that do not produce marks on the hard copy. Some of these machine functions, such as space, upper and lower case, and tabulate, relate to ordinary typing practice. Other functions are specific to the tape-punching and reading features of the Flexowriter and are described in the manufacturer's instruction manual.

The Flexowriter punches 8-channel paper tape that is 1 inch wide. Five channels appear on one side of the sprocket holes, and three channels appear on the other side. In accounts of the Flexowriter, the eight channels in the tape are numbered consecutively from 1 to 8, going from left to right when the tape is arranged with five channels on the left of the sprocket holes. The sprocket holes therefore appear between channels 5 and 6 in this numbering.

For programming purposes, it is convenient to denote the pattern of holes in a row of tape by a number that is constructed in the following way. A "binary value" is given to each channel in accordance with the scheme

Channel Designation	1	2	3	4	5	6	7	8
Binary Value	128	64	32	16	8	4	2	1

The number that represents a punched row of tape is formed by adding the binary values of all the channels that contain holes in the row of tape that is under consideration. This number is called the "left-to-right binary" representation of the tape code, because it is, in programming terminology, the number whose binary representation is formed by the pattern of holes in the tape reading from left to right (with the tape positioned so that five channels appear on the left of the sprocket holes). Its use is in accordance with the "read" instructions of many computers that read paper tape directly. It is consistent, too, with the instructions that read other input media, and the conventions of equipment that copy codes from paper tape to these other input media.

2.3 Processing Paper Tape for Input

The C.C.L. 709 was not equipped with attachments to read or punch paper tape directly. Information on Flexowriter tape was trans-

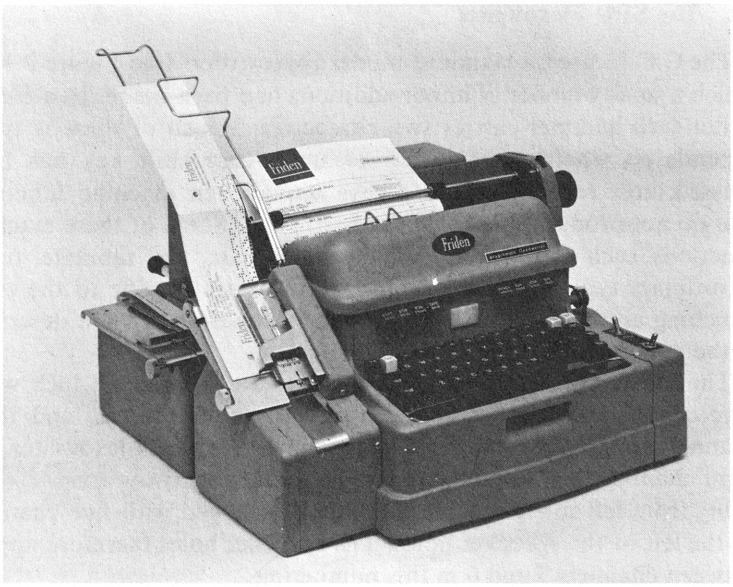

Friden
Friden
Friden
Friden

Figure 2.1. *Friden Flexowriter* automatic writing machine, model SPD*

*A Trademark of Friden, Inc.

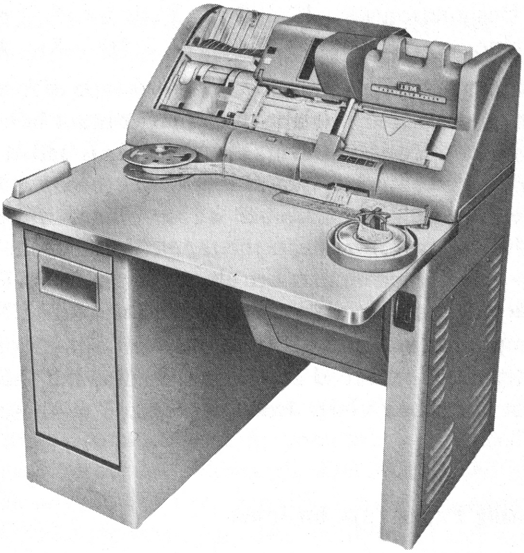


Figure 2.2. *IBM 047 card-to-tape converter.*

ferred to punched cards in a preliminary off-line operation for input to the computer. Photon operating tapes similarly were punched, in an off-line operation, from cards that were generated as computer output. In the operating practice adopted in the C.C.L., the bit images of successive rows of Flexowriter tape were punched in successive columns on a deck of cards mechanically. A distinguishing 7–9 code was punched in column 1 of every card, and Flexowriter codes were punched in columns 2 through 72. Only rows 9 through 2 of the card were used. The conversion was performed on an IBM 047 paper tape to punched-card converter[2] which had been modified by M.I.T. personnel to punch bit images of the Flexowriter codes (see Figure 2.2). This use of binary card images was the continuation of a practice which had been adoped in the early days of the project. It would have been far better for many reasons to have used the IBM 047 in the standard manner to punch Hollerith cards (cards using the standard IBM code, called Hollerith code) from the Flexowriter tape. This matter is discussed further in Section 6.2. In the C.C.L. experiments, however, binary cards were used, and the Flexowriter code images in them were written on magnetic tape in 71 word records off-line for input to the programs.

2.4 The IBM 709 Computer

The IBM 709 computer[3] in the C.C.L. was equipped with a 32 K core store, 12 magnetic tape units, and a cathode-ray display. Almost all the programs were written in the FORTRAN II language.[4] The rest were coded in FAP.[5] Several standard utility subroutines issued by IBM were used. The programs ran under the standard IBM distributed Fortran Monitor System.[6] The compiled programs also were run on the IBM 7090 computer without modification.

2.5 Preparing Paper Tape from Computer Output

Photon codes were written by the programs on magnetic tape for subsequent off-line preparation of 8-channel punched paper tape. Three methods were used at various times for this latter purpose. The only method that was at all satisfactory for bulk operation used the IBM 1401 and 1012 attachment at the M.I.T. Lincoln Laboratory to punch paper tape directly from suitably written magnetic tape. This equipment is described fully in manuals that are provided by its manufacturer.[7]

Prior to the availability of the tape punch at the Lincoln Laboratory, use was made of a Friden tape punch that was attached to the IBM 047 machine by some improvised circuitry. This enabled paper tape to be punched with codes that were copied from successive columns of cards which were read by the 047 machine. This procedure was considerably slower than the 1401–1012 punch operation and subject to frequent machine errors.

Use also was made at one juncture of a Friden punch attached to an IBM 722 unit by some further improvised circuitry. This was quite unsatisfactory, however, and its use was discontinued.

2.6 The Photon 560 machine

The Photon 560 machine (see Figure 2.3) consists of a type 500 photographic unit and a type 560 control unit.[8] These are housed in separate cabinets which are joined by a connecting cable. The photo-

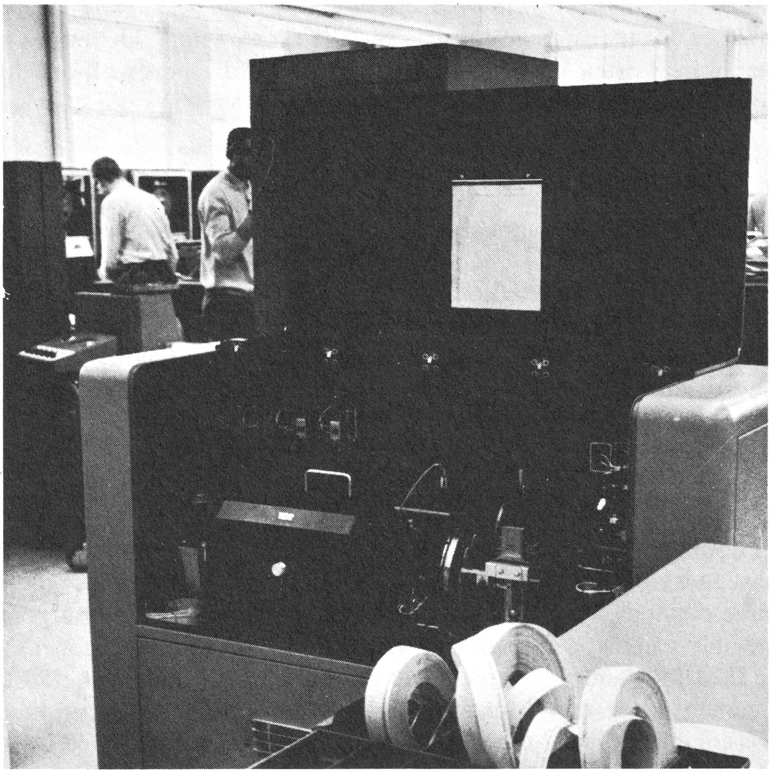


Figure 2.3. *C.C.L. Photon 560 machine with IBM 709 computer in background.*

graphic unit is very similar to that in other Photon disc phototypesetting machines. To run the Photon 560 machine, the operator turns on the main line switch, loads a paper tape in the tape reader that forms part of the control unit, and turns on the stop-start switch. The tape is then read mechanically, row by row, and the control unit converts the codes in the tape into signals which are sent through the connecting cable to the photographic unit. At convenient intervals, the machine is stopped by the operator, the film magazine is taken to a darkroom, and the film is developed.

The main features of the photographic unit are the flash lamp, the character disc, the lens turret, the reflecting prism, and the film magazine (see Figure 2.4). These are identical with the corresponding components in the other models of Photon disc phototypesetting machines. The main features of the control unit are the paper tape reader and the console. These various components are discussed in turn in the Sections 2.7 to 2.15 which follow. The basic relationships of the Photon 560 machine and the more familiar tape-operated Photon disc machines are as follows. The 560, like the 540, is operated by 8-channel tape. This sets film in the direction in which it can be read after developing on the 560, but in the reverse direction on the 540. Each 560 code occupies 2 tape frames, whereas each 540 code occupies 3 tape frames. The 560 tape contains spacing data in basic machine units. The "automated set card" attachment to the 200 enables 6-channel tape, without escapement data, to be read in the forward direction. The 513 machine reads 6-channel tape, with much the same information, backward.

2.7 The Flash Lamp and the Character Disc

The photographic unit (see Figure 2.4) houses a glass disc which is 8 inches in diameter. Photographic negatives of 1440 characters are arranged in 8 concentric rings on the surface of the disc. Each ring contains 180 uniformly spaced characters. The disc rotates in a vertical plane, in front of a narrow metal tube that contains a small flash lamp. This tube is mounted near the rim of the disc. The disc rotates about a spindle which is mounted in a small cradle. This cradle can occupy 8 parallel positions. Changing the position of the cradle moves the axis of rotation of the disc by a short distance in the vertical plane. During one rotation of the disc, each character on a single ring moves in turn past an opening in the tube which contains the flash lamp. The position of the cradle determines which ring of characters moves past the tube opening. The cradle is moved by appropriate codes in the

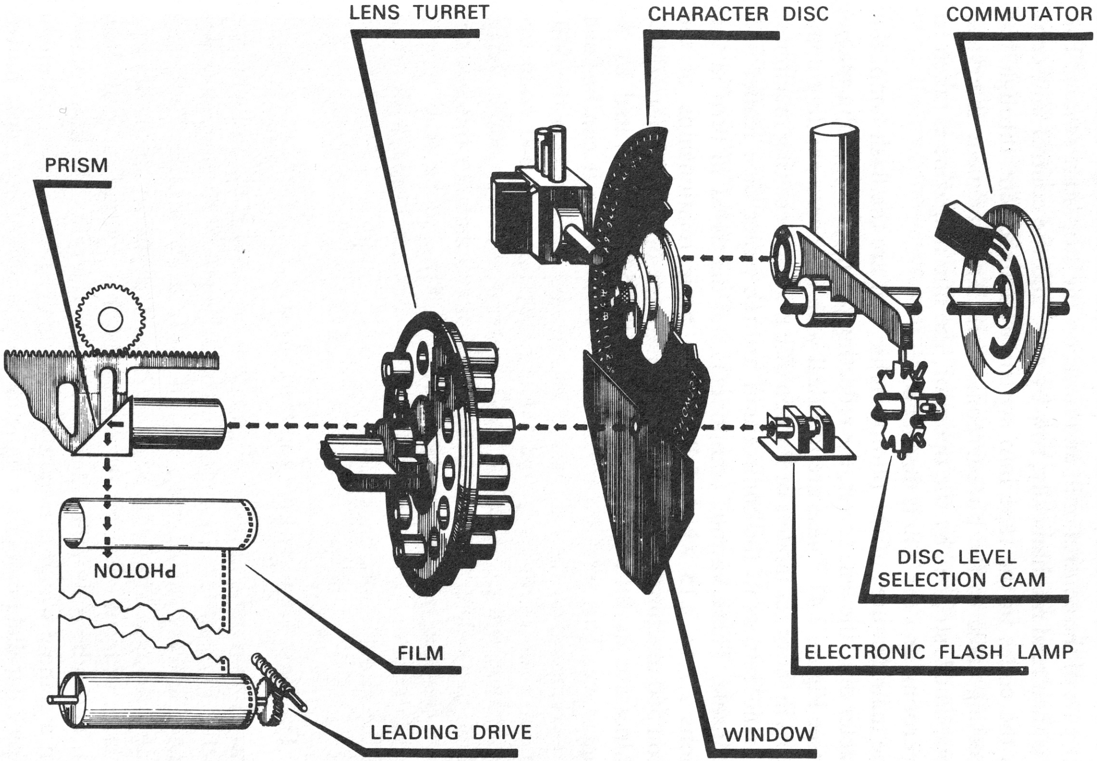

LENS TURRET
CHARACTER DISC
COMMUTATOR
PRISM
PHOTON
DISC LEVEL SELECTION CAM
FILM
ELECTRONIC FLASH LAMP
LEADING DRIVE
WINDOW

Figure 2.4. *Optical system of the Photon 560 machine.*

operating tape. The flash lamp emits bursts of illumination that last 3 to 5 microseconds. These are emitted at times which are determined by signals from the control unit and which are related to the movement of the disc. Since the disc rotates at 600 revolutions per minute, it is effectively stationary during the period of illumination of the flash lamp. Each burst of light is timed to illuminate the character which is specified by the code that has just been read in the operating tape. This code is called a character sequence code.

Different discs that provide different selections of type faces and special symbols may be used in separate runs. Discs are changed by a trivial manual operation. Several hundred discs are in use in installations of Photon 200, 513, and 540 machines in the United States and elsewhere. Each disc is prepared by a routine factory operation. In this operation, 1440 characters that are drawn on separate cards are copied photographically onto the disc. The cards on which the characters are drawn can be selected from an existing library of some 300 styles of type and some 3500 special symbols, which may be supplemented by drawing new characters if necessary.

For operational purposes, each ring of characters on a disc is considered in two halves. These are called disc levels. They are identified in the disc specifications by the numbers 1 through 16. The characters on each disc level are identified by sequence numbers that take values 1 through 90. The disc levels and sequence numbers of all the characters on a disc are listed on the specification chart of the disc that is called its C-sheet.

2.8 The Lens Turret

After passing through the disc, the light beam from the flash lamp passes through a lens (see Figure 2.4). This magnifies the illuminated character. A total of twelve lenses of different magnifying powers are mounted in a vertical plane, around the edge of a circular lens turret. This turret can be rotated about a horizontal axis, to bring any one of the lenses into the path of the light beam that has passed through the disc. The movement of the lens turret is controlled by codes in the paper tape that are called lens turret codes. The turret remains stationary until a lens turret code moves it to bring another lens into operation. Further operations are then delayed automatically until the movement of the turret is complete.

A wide range of magnifying lenses can be used in the lens turret. Each of these lenses is mounted in a tubular container by the factory. The containers are of uniform size, so that any lens can occupy any turret station. The set of lenses in the turret can be changed by the operator, but these changes do require a rather careful adjustment.

2.9 Character Alignment

Normally, the lenses are mounted so that their centers all lie on a circle whose center is on the axis of rotation of the lens turret. In general, a line of type is set so that the lowest points of all the characters, except the letters g, j, p, q, and y, which are called descenders, touch a common line. The line through any character that falls on this common line is called the base line of the character. Also, in a line of type, the highest points of all the characters, except the letters b, d, f, h, k, l, and t, which are called ascenders, touch a common line.

In general, each character is placed on a Photon disc so that when it is illuminated by the flash lamp, its base line projects onto the horizontal center line of the lens in current use. The position of the base line in the image of a character that is formed by this lens is therefore independent of the choice of character and of the identity of the lens. Sometimes use is made of characters whose negatives on the Photon disc are displaced slightly from those of normal characters, so that their images are formed with subscript and superscript alignments, relative to the images of characters which are positioned normally on the disc. Lenses that are displaced slightly in their mounting from the normal lens position are also used at times to form images in subscript and superscript positions on the film, from characters which occupy normal positions on the disc.

2.10 Type Size

All the negatives on the Photon disc are of commensurate size, so that the images which are formed by a single lens correspond to a single size of type. The terms that are used to describe the dimensions of printed characters, and to describe vertical and horizontal spacing in phototypesetting, are based on conventional typesetting practice. This makes use of pieces of metal type of rectangular cross section. Each piece of type has a projection which can be seen to have the shape of a character in reverse when it is viewed end-on. Conventionally, a character is printed by inking this projection, or a molded copy of it, and then bringing the inked relief of the character into contact with paper, to which particles of ink adhere and form the appropriate visible mark.

Pieces of type are stocked in certain standard sizes, the size being measured by the height in "points." One point is $\frac{1}{72}$ inch. In general, a line of type is a molded copy of pieces of type that are of uniform height and embossed with characters that are commensurate with each other. The base lines of the characters on pieces of type of the same point size are collinear when the lower edges of the pieces of type are collinear. The height and width of a piece of type, rather than the distances between portions of a visible character, are used to describe the character's dimensions. The distance between the neighboring edges of two consecutive pieces of type in a line of type provides a simple and meaningful measure of horizontal spacing. The distance between the base lines of two consecutive lines of type, however, is used most frequently as the measure of vertical spacing. This distance is called body lead. For minimum vertical spacing of type of uniform size, the body lead is equal to the point size of the type. Horizontal and vertical spaces are measured in points and picas. A pica is a sixth of an inch, i.e., twelve points.

The description of letter size in terms of "point size" is retained for phototypeset material. By definition, the point size of a phototypeset character is the point size of a character of the same size and style that has been printed from metal type. The lenses in the Photon machine are identified by the point sizes of the characters which they produce from the negatives on the disc.

Standard lenses can be used on Photon machines to provide the following point sizes in the printed output: 5, 6, 7, 8, 9, 10, 11, 12, 14, 18, 24, 30, 36, 42, 48, 60, 72. Lenses also can be obtained to give other point sizes in the range 5 to 72. Any of these lenses can be mounted to project characters into subscript and superscript positions.

2.11 The Reflecting Prism

After passing through the magnifying lens, the light beam passes through a prism. This deflects the beam by an angle of 90° in a horizontal plane. The light beam then passes through a long narrow window in the side of the film magazine. The long edges of the window are horizontal. The window exposes a strip of a roll of film that is mounted between horizontal rollers. One roller is above the plane of the light beam, the other is below it. The light beam impinges on the film and forms a photographic image of the character on the disc which is illuminated. After a character is illuminated and recorded photographically in this way, the prism carriage moves a small distance horizontally, along a path that is parallel to the window of the magazine. The images of the characters that are illuminated in successive cycles of the operation are thus projected onto successive portions of the strip of film which is behind the window. The distance that the prism moves after a character has been projected is called the escapement of that character. It is determined by a signal from the control unit. This signal controls the spacing between characters on the film. The type 560 control unit derives the signal from a code in the paper tape. The code is called an escapement code. In general, a character sequence code on a Photon 560 operating tape is followed by an appropriate escapement code.

When pieces of metal type carry the same character in the same style, but in different sizes, the amount of space that is left on each side of the character, as well as the size of the character itself and the width of the piece of type, are proportional to the point size. The ratio of width to height of a piece of type thus is a constant for a particular character in a particular style. The number that is obtained by multiplying this ratio by 18 is called the relative width of the character, in the style of type that is under consideration. It transpires that for a majority of type faces, the relative widths which are defined in this way are integers for all letters, digits, punctuation marks, and special symbols. In general, the distance by which the Photon prism is moved after a character has been set within a word is just the width of the piece of metal type on which the character would have been provided in hot metal work. This distance is calculated for each character that is set on the Photon 560 machine by the computer program which constructs the sequence of codes that is punched in the operating tape. Each character on a Photon disc has a relative width that is determined by its typographic style. The table of relative widths of the disc that is to be used is stored in the computer. In general, each escapement is calculated basically as the product of its relative width and the point size in current use.

The manufacturer provides a separate specification chart, called a D-sheet, for each of the sixteen levels of a disc. Each D-sheet lists the relative widths of the ninety characters on the disc level to which it pertains.

It can be seen that if type is set in a point size that is an integer and if characters have relative widths that are integers, then escapements are always multiples of $\frac{1}{18}$ point. Now 1/(18x72) inches is very close to 2^{-9} centimeters, and for compatibility with the metric system, Photon escapements are controlled in multiples of the "basic machine unit" of 2^{-9} centimeters. The point is redefined for Photon work as 18 basic machine units, that is, as $\frac{9}{256}$ centimeters, or 1.0025 English points.

It should be mentioned that there are some further details of spacing that affect Photon work. The horizontal spacing increment which is possible on the Photon machine in fact is 2 basic machine units, that is, $\frac{1}{256}$ centimeters. In some type faces, not all the characters have integer relative widths. The other characters in these faces have relative widths that are half-integers. Moreover, for some fonts, particularly those that are boldface, the relative widths that are needed to avoid an overcrowded appearance of print are slightly greater than integers and half-integers. It is expedient, then, to change the definition of relative width slightly, to be a property of each character in each style of type, so that it is an integer or a half-integer which is multiplied by a factor called the set size, to compute escapement. The set size factor depends on the style of type and point size in use. For most faces, set size is equal to point size, but for some boldfaces, set size is obtained from the point size by adding $\frac{1}{2}$ or 1 according to the typographer's preference. The PC6 programs did not take this possible difference of set size and point size into account, and they deal with relative widths and point sizes as these were first described in this section.

An unpunched row of tape allows the disc to rotate without the illumination of any of the characters. An unpunched row of tape followed by an escapement code thus provides extra space in the developed output film. Space for indention, space between words, and space between columns of material can be represented in the operating tape in this way. The computer program that produces the Photon codes must determine these spaces, by reference to format specifications, and to alignment and justification requirements.

2.12 The Film Magazine

The prism, which deflects the image of the illuminated characters onto the film in the magazine, can travel up to about 9 inches from its starting position, as the result of the execution of a succession of escapement codes. A further code, that is called the carriage-return code, is used to end a line of output. This code restores the prism to its starting position. When the prism is in this position, it projects the beam of light that has passed through the disc onto a point that is .54 of an inch or more from the left edge of the film depending on the position of an adjustable stop. The phototypeset lines can therefore be up to about $8\frac{1}{2}$ inches long.

The phototypeset output of the Photon machine is spaced vertically by the rotation of the rollers that carry the film. The movement of the rollers is controlled by further codes in the operating tape. These codes must be punched by the computer programs which produce the tape. The illumination of characters is interrupted while the rollers are in movement. The rotation of the rollers causes a new strip of film to be exposed behind the window of the magazine, so that a new line of characters may be recorded photographically at an appropriate distance below the previous line.

The amount that the rollers turn, and hence the vertical spacing between successive lines of output, can be controlled by several codes. The Photon 560 control unit contains a buffer that can store any integer in the range 1 to 49. This is called the leading buffer. When a switch on the control unit, called the leading switch, is turned off, the contents of the leading buffer is set by the operator on a set of buttons. When the leading switch is turned on, the contents of the leading buffer are reset whenever a "normal leading" code is read in the operating tape.

The film in the magazine is spaced vertically whenever a carriage return code is executed. The distance in points that the film is spaced is equal to the contents of the leading buffer at that time. This vertical spacing of the film follows the return of the prism to its starting position. Both operations are produced automatically by the carriage return code. The prism can be restored to its starting position, without any movement of the film, by a further code in the operating tape. This code is called the zero leading code. The film is spaced vertically, but without any change in the position of the prism, when any one of four further codes, that are called additional leading codes, is read. These four codes move the film by 1, 2, 4, and 8 points, respectively.

2.13 Developing Output

Phototypesetting film for the photographic unit is available commercially, in rolls which are 50 and 100 feet long, and 105 to 254 mm. wide. Runs may be interrupted at convenient reference points to process batches of the film. It is customary, for example, to program several vertical space codes in the control tape to leave a vertical gap in the printing wherever the film is to be cut into pages. The "page breaks" in the tape can then be recognized by an operator. A counter on the console records the length of film which is used in a run. The Photon machine therefore can be interrupted by the operator in accordance with criteria of several types. After setting the start-stop switch to stop, the film magazine is lifted from the photographic unit, and transferred to a darkroom. The magazine is opened, and the exposed film on the winding roller is cut from the unused film and removed for developing. The unused film is advanced and attached to the winding roller, and the magazine is closed and returned to the photographic unit for further use. An attachment to the magazine punches two small perforations in the edge of the film whenever a code, called the film punch code, is read in the operating tape. It is convenient to include this code in the tape between the codes for successive pages of output. The operator working in the darkroom can then gauge where the film may be cut into pages.

The character image is transmitted in the following manner. The negatives of characters on the Photon disc are on the surface that is closer to the lens, that is, away from the flash lamp. The negatives appear as readable transparent characters on an opaque background if viewed through the other surface of the disc. They appear reversed (mirror images) if viewed through the surface closer to the lens. The top of each character points radially inward, and the character that at any instant is in front of the flash lamp points vertically upward at that time. The lens reverses the image about a vertical line and also inverts it. The prism reverses the image again about a vertical line so that if it is viewed through a ground glass screen, it appears inverted and reversed, that is, as if it were on a page of print that was upside down. Film is mounted in the photographic unit with the emulsion facing the window. A positive print of the characters which have been projected is formed when the film is developed, since the disc carries negatives. This print is the inverse (reflection about a horizontal line) of readable type, so the winders move the film downward in the magazine before each line of characters is projected. If the film is turned upside down after it has been developed, the print on it is the reverse (mirror image) of readable type when the viewer is in front of the surface that was emulsified. For this reason it is called wrong-reading. Since the film that is usually employed has a transparent base, the developed film can be read by viewing the developed image through the base, i.e. from the side opposite that which was emulsified. Wrong-reading positives are the most useful form of output from phototypesetting machines for certain photolithographic processes.

Right-reading positives can be produced, when necessary, by mounting a further optical component that inverts each character image, between the lens turret and the prism that controls the horizontal spacing. This produces a right-reading but uninverted image in the emulsion, and the direction of the film movement between lines is reversed when right-reading output is produced in this manner. Sensitized paper can be used instead of film when the photographic unit is operated in the right-reading mode, to produce readable type on an opaque background.

Wrong-reading negatives, suitable for surface plates for offset lithography also can be produced as Photon output, when the photographic unit is operated in the wrong-reading mode. To do this, the image is reversed in the developing process by bleaching to remove the latent image, fogging to decompose the emulsion that has not been illuminated in the Photon, and fixing to retain this as the black portion of the negative.

2.14 Pi-Mat

Eight extra characters or ciphers, in addition to those on the disc, can appear in the output of a continuous run of the Photon 560 machine if the optional "pi-mat" feature is included in the photographic unit. A 42-point negative of each character is first prepared on a separate piece of film that measures 1½ inches by 2 inches. A library of these pi-mat films is available. Further pi-mat films can be produced by a simple factory operation or in any photographic laboratory. Each film is inserted in a separate frame in the pi-mat attachment. There are eight of these frames in the attachment. The character in a pi-mat frame is illuminated, and its image is projected onto the film in the magazine, when an appropriate code is read in the operating tape. The pi-mat attachment contains a lens which diminishes the image of the illuminated character, so that the image on the output film is in the point size which is characteristic of the lens that is in current use.

2.15 The Operating Tape

The Photon 560 machine reads 8-channel paper tape that is one inch wide and which has five channels on one side of the sprocket holes and three channels on the other side. The tape is thus similar to the Flexowriter tape, but the codes are completely different. In discussions of the Photon machine, the eight channels in the tape are designated S, A, B, C, D, E, F, G, respectively, from left to right when the tape is arranged with five channels on the left of the sprocket holes, which occur therefore between the D and E channels. It is convenient to denote a Photon control code by the letters that name the channels which are punched to form the code. For example, the carriage-return code is denoted by SAC. This means that the first, second, and fourth positions in the tape are punched for a carriage return. For programming purposes it is convenient to denote the pattern of holes in a row of Photon tape by a number that is formed by adding the binary values of the channels which are punched, where the binary values of the individual channels are as follows:

Channel Designation	S	A	B	C	D	E	F	G
Binary Value	128	64	32	16	8	4	2	1

As with the Flexowriter tape, this number is called the left-to-right binary value of the code.

2.16 The Control Codes

Two successive rows of tape are read by the Photon 560 control unit for each operation of the photographic unit. Two rows that are considered together in this way are called a doublet. Blank doublets, that is, pairs of blank rows, may be left in the tape and are ignored. The code that is punched in the first row of a doublet determines the operation that the machine performs. Ninety character sequence codes are used for selective illumination of the individual characters on the disc level code. Some forty-five operating codes are used to control the other machine functions. The operating codes are listed in the manufacturer's literature.

A standard set of ninety commonly occurring characters is provided in a standard order on one or more levels of many discs. This standard arrangement includes lower- and upper-case letters, digits, punctuation marks, and certain combination characters. The sequence numbers of the letters and digits in this standard arrangement often are retained on disc levels that also contain special characters in the other positions.

The second row of every doublet, except null, horizontal space, sequence number, and normal leading doublets, is blank. A null doublet starts with a delete code (all channels punched) and is ignored, regardless of what is punched in the second row.

The second row of horizontal space and sequence number doublets contains an escapement code. A single escapement code can advance the position of the prism by $\frac{n}{256}$ centimeters, where n is less than 128 and is punched in "right-to-left binary," that is as a binary number with the units position on the left of the tape. The integer that is punched in left-to-right binary to provide a given escapement represented in right-to-left binary can be found by a trivial arithmetic process or by reference to a table. Escapements were represented in this way as a consequence of the decoding circuits in the Photon machine. These circuits could be modified easily to allow left-to-right representation of the escapements.

For typographic purposes, distances are usually measured in points.

Since 1 point $= \frac{9}{256}$ centimeters, an escapement of k points is obtained by punching $9k$ in right-to-left binary in the control tape. An escapement greater than $\frac{127}{256}$ centimeters is produced by punching two or more doublets in succession.

The second row of a normal leading doublet consists of the integer to be stored in the leading buffer, punched in right-to-left binary form. This integer is the distance in points by which the film is moved when a carriage-return code is executed.

A special device may be attached to the control unit, to check the synchronization of the tape reader. The computer programs that generate the paper tape must then provide a punch in the G channel of the second row of every doublet regardless of whether or not the rest of the row is blank.

3. Preparing TYPRINT Source Material

3.1 The TYPRINT Option

When a manuscript is prepared for publication, the editor, and at times the author, use the margin to write instructions to the typesetter. These instructions tell the typesetter the line length and the size and style of type that should be used, the portions of the text that should be centered or indented, the special symbols that appear, and any other items of information which are needed to set type from the manuscript. The TYPRINT option of the PC6 system of programs dealt with material typed on a Flexowriter from manuscripts in which the instructions, conventionally written in the margin, were placed instead between square brackets at appropriate positions in the text, and in a suitably coded form. The way that the codes for these instructions were made up is explained in Sections 3.2 to 3.14. The Flexowriter copy was proofread, and if any errors were found, the tape was corrected by the processes that are described in Sections 3.15 and 3.16. The paper tape that had been punched by the Flexowriter and corrected, if necessary, then was used as input to a computer process. The way that this was done and the way that the computer produced an operating tape for the Photon machine are described in Sections 3.17 and 3.18, but it was not necessary for the personnel who did the keyboard work, or the "annotators," that is, the personnel who selected and inserted the control codes into the manuscripts, to concern themselves with these matters if they did not wish to do so.

It should be stressed that the TYPRINT conventions were essentially experimental as regards the selection of typographic style and format characteristics which they enabled the annotator to control, and as regards the way in which they allowed the requisite characteristics to be specified. Alternative and extended ways of defining typographic requirements in the input to computer-aided typesetting processes form the basis of much of this monograph. It should be stressed too that the TYPRINT option that is described in this chapter would not offer economic advantages for many typesetting jobs. The TYPRINT option is described first as it provides a convenient introduction to the further ways of controlling typography that are considered in later chapters. Moreover, reading this chapter will serve a very useful purpose if it provokes a reader, for whose typesetting problem the TYPRINT conventions are unsuitable, into specifying the conventions that really would suit his job.

Throughout the present chapter, statements concerning the preparation of Flexowriter source material or the operation of the computer all carry the implicit qualification, "within the TYPRINT conventions," and do *not* apply to computer-aided typesetting in any wider sense. The examples that are included to illustrate the usage of the TYPRINT codes were run soon after the C.C.L. Photon machine was installed, and contain some flaws as a consequence of minor malfunctions of the photo unit and minimal photographic developing procedures. These examples do not do justice to the quality of work that is possible with the Photon equipment, and regularly achieved by the adoption of adequate maintenance and quality control procedures. The examples all were run with disc 78, which is rather below the standard of disc that is generally obtained, as regards the positioning of the characters. The disc was obtained at a discount for this reason.

3.2 Styles of Type

When a Photon machine is used to set a job, a disc must be selected first from those that are available to the user. This is the case for Photon machines such as the 200 and 540 models that are in relatively widespread use, as well as for the 560, which the author used and which is almost unique. In principle a disc is chosen to provide the fonts and special characters that the typographer requires. In practice, a limited number of discs is available in any Photon installation, and the typographer selects a set of type faces that meet his general needs and are all on a single disc. Two discs were used in the C.C.L. Disc 78

Table 3.1. *Type faces on disc 78.*

Name	Alphabet	Code
Univers extrabold	**abcdefghijklmnopqrstuvwxyz1234567890** **ABCDEFGHIJKLMNOPQRSTUVWXYZ**	DL1
Univers medium	abcdefghijklmnopqrstuvwxyz1234567890 ABCDEFGHIJKLMNOPQRSTUVWXYZ	DL2
Univers bold	**abcdefghijklmnopqrstuvwxyz1234567890** **ABCDEFGHIJKLMNOPQRSTUVWXYZ**	DL3
Century schoolbook	abcdefghijklmnopqrstuvwxyz1234567890 ABCDEFGHIJKLMNOPQRSTUVWXYZ	DL4
Century bold	**abcdefghijklmnopqrstuvwxyz1234567890** **ABCDEFGHIJKLMNOPQRSTUVWXYZ**	DL5
Scotch	abcdefghijklmnopqrstuvwxyz1234567890 ABCDEFGHIJKLMNOPQRSTUVWXYZ	DL6
Granite	abcdefghijklmnopqrstuvwxyz1234567890 ABCDEFGHIJKLMNOPQRSTUVWXYZ	DL7
Highland	abcdefghijklmnopqrstuvwxyz1234567890 ABCDEFGHIJKLMNOPQRSTUVWXYZ	DL8
Univers med. italic	*abcdefghijklmnopqrstuvwxyz1234567890* *ABCDEFGHIJKLMNOPQRSTUVWXYZ*	DL10
Univers bold italic	***abcdefghijklmnopqrstuvwxyz1234567890*** ***ABCDEFGHIJKLMNOPQRSTUVWXYZ***	DL11
Century school italic	*abcdefghijklmnopqrstuvwxyz1234567890* *ABCDEFGHIJKLMNOPQRSTUVWXYZ*	DL12
Century bold italic	***abcdefghijklmnopqrstuvwxyz1234567890*** ***ABCDEFGHIJKLMNOPQRSTUVWXYZ***	DL13
Scotch italic	*abcdefghijklmnopqrstuvwxyz1234567890* *ABCDEFGHIJKLMNOPQRSTUVWXYZ*	DL14
Granite italic	*abcdefghijklmnopqrstuvwxyz1234567890* *ABCDEFGHIJKLMNOPQRSTUVWXYZ*	DL15
Highland italic	*abcdefghijklmnopqrstuvwxyz1234567890* *ABCDEFGHIJKLMNOPQRSTUVWXYZ*	DL16

Table 3.2. *Type faces on disc 77.*

Name	**Alphabet**	**Code**
Scotch	abcdefghijklmnopqrstuvwxyz1234567890 ABCDEFGHIJKLMNOPQRSTUVWXYZ	DL1
Bodoni bold	**abcdefghijklmnopqrstuvwxyz1234567890** **ABCDEFGHIJKLMNOPQRSTUVWXYZ**	DL2
Bodoni	abcdefghijklmnopqrstuvwxyz1234567890 ABCDEFGHIJKLMNOPQRSTUVWXYZ	DL3
Bodoni small caps	ABCDEFGHIJKLMNOPQRSTUVWXYZ	DL6
Techno bold	**abcdefghijklmnopqrstuvwxyz1234567890** **ABCDEFGHIJKLMNOPQRSTUVWXYZ**	DL7
Scotch italic	*abcdefghijklmnopqrstuvwxyz1234567890* *ABCDEFGHIJKLMNOPQRSTUVWXYZ*	DL9
Bodoni bold italic	***abcdefghijklmnopqrstuvwxyz1234567890*** ***ABCDEFGHIJKLMNOPQRSTUVWXYZ***	DL10
Bodoni italic	*abcdefghijklmnopqrstuvwxyz1234567890* *ABCDEFGHIJKLMNOPQRSTUVWXYZ*	DL11

carried the type faces which are displayed in Table 3.1. It was the property of the C.C.L. Disc 77 carried the type faces which are displayed in Table 3.2. It was loaned to the C.C.L. by the Machine Composition Company.

3.3 Sizes of Type

The C.C.L. Photon machine was equipped with lenses to provide type in 6, 8, 9, 10, 11, 12, 14, 18, 24, and 30-point sizes. Two further lenses provided 6-point subscripts and superscripts to characters that were set in 10-point type. The alphabet is displayed in the sizes that were available on the C.C.L. Photon in Table 3.3. If other lenses had

Table 3.3. *Type sizes available with the C.C.L. Photon machine.*

Point size	**Alphabet**	**Code**
6	abcdefgHIJKLMNOPQRSTuvwxyz	LS6
8	abcdefgHIJKLMNOPQRSTuvwxyz	LS8
9	abcdefgHIJKLMNOPQRSTuvwxyz	LS9
10	abcdefgHIJKLMNOPQRSTuvwxyz	LS10
11	abcdefgHIJKLMNOPQRSTuvwxyz	LS11
12	abcdefgHIJKLMNOPQRSTuvwxyz	LS12
14	abcdefgHIJKLMNOPQRSTuvwxyz	LS14
18	abcdefgHIJKLMNOPQRSTuv	LS18
24	abcdefgHIJKLMNOPQ	LS24
30	abcdefgHIJKLMN	LS30

been substituted in the lens turret, the PC6 programs could have been adapted very easily to deal with their point sizes.

3.4 Setting a Single Paragraph

To set a single paragraph of text in a single style and size of type, the paragraph was typed on a Flexowriter with three control codes at the beginning and one code at the end. One of the codes at the beginning signaled the beginning of a new job to the computer. Another code specified the style of type that was to be used. The third code specified the size the type should be. The code at the end of the text signaled the end of the job to the computer.

To prepare a paragraph for setting with disc 78 on the C.C.L. Photon, one of the styles of type in Table 3.1 and one of the sizes of type in Table 3.3 were first selected. A left square bracket was typed to show that the characters which followed constituted codes, and not text that was to be set. The code IN then was typed to show that the job was starting, and a comma was then typed because another code followed. The code for the selected style was typed next. This code was copied from the third column of the line of Table 3.1 that contained the alphabet which was to be used. For example, if the output was to be set in Univers Bold, DL3 was typed. For output to be set in Scotch, DL6 was typed. This code was followed by a comma and then the code for the size of type that was required. This latter code was copied from the third column of the line of Table 3.3 that contained the size of alphabet which was to be used. For example, the code LS6 was copied to obtain 6-point type, and LS12 for 12-point type. A right square bracket then was typed to show that text matter followed, and the carriage return key was pressed. The first word of the paragraph was indented by using the space bar. Then the paragraph was typed just as if it were being typed on an ordinary typewriter, except that words could not be broken at the end of a line. The code EN was typed between square brackets at the end of the paragraph to instruct the computer that the end of the material which was to be set had been reached. This completed the work that the keyboard operator had to do. It differed from ordinary copy-typing practice only in the inclusion of the control codes and the use of the space bar to indent. Some more rules had to be known to use the tabulate key in an appropriate manner. These rules are explained in Section 3.5.

The paper tape that was produced by the Flexowriter was given to the computer operating staff who used it as input to a process that produced an operating tape for the Photon machine. This machine in turn produced a sheet of paper or a sheet of film, on which the paragraph was set in the style and size of type that was specified by the DL and LS codes. This film or paper could be used as the original for lithographic reproduction or photocopying. The programs automatically set the line length to 432 Photon points. This is a little more than 6 inches, because a Photon point is a little longer than an English point, as explained in Section 2.11. The line length could be changed by the user, as explained in Section 3.6. The computer left the minimum space between successive lines unless additional spacing was requested by the user. The way that this was done is described in Section 3.7. The programs set the page length automatically to 540 Photon points. This is a little more than 540 English points, which equals 7½ inches. The way that the user could change the page length is described in Section 3.10.

The instructions that have just been given can be illustrated by a simple example. Suppose that a one-paragraph comment about this section was to be printed in 8-point Scotch type. The Flexowriter was used to type the paragraph, with appropriate codes at the beginning and end. The Flexowriter copy and printed Photon output are shown as Example 3.1.

The computer determined where the lines should end, and put extra space between the words wherever necessary to make the right-hand margin straight, that is, to "justify" the text. The last line of a paragraph was left unjustified. A word that ended a line of the Flexowriter input did not necessarily end a line of the printed output. The computer recognized the end of a line of input and provided the appropriate space in the output. In the typographic practice that was recommended to the C.C.L., this consisted of two spaces after a period and one space after any other character. Different spacing conventions and hyphenated words in the input could have been accommodated by relatively simple extensions of the program as discussed in Section 7.4.

A paragraph could be set on the C.C.L. Photon with disc 78 in 150 different ways, since there were 15 different styles of type on the disc and each of these could be printed in 10 different sizes on the C.C.L. equipment. Example 3.1 used 8-point Scotch type. To use 10-point Highland type, the Flexowriter was used to type the paragraph, with [IN, DL8, LS10] at the beginning and [EN] at the end. The Flexowriter copy and the printed Photon output are shown as Example 3.2.

In the codes that selected type faces, the letters DL stood for disc level. The code DL1 selected the font on disc level 1, the code DL2 selected the font on disc level 2, and so forth. Similarly in the codes

```
[IN,DL6,LS8]
     A very simple example of a TYPRINT application is the
production of a single paragraph of printing, in a single
size and style of type.   Any one of 16 different styles can
be used.   The alphabet is printed in each of these styles in
table 1, which gives the name of each style and also a code
that will be used later.   [EN]
```

A very simple example of a TYPRINT application is the production of a single paragraph of printing, in a single size and style of type. Any one of 16 different styles can be used. The alphabet is printed in each of these styles in table 1, which gives the name of each style and also a code that will be used later.

Example 3.1. *Flexowriter copy and 8-point Scotch TYPRINT output.*

```
[IN,DL8,LS10]
     A very simple example of a TYPRINT application is the
production of a single paragraph of printing, in a single
size and style of type.   Any one of 16 different styles can
be used.   The alphabet is printed in each of these styles in
table 1, which gives the name of each style and also a code
that will be used later.   [EN]
```

A very simple example of a TYPRINT application is the production of a single paragraph of printing, in a single size and style of type. Any one of 16 different styles can be used. The alphabet is printed in each of these styles in table 1, which gives the name of each style and also a code that will be used later.

Example 3.2. *Flexowriter copy and 10-point Highland TYPRINT output.*

that selected point size, the letters LS stood for lens size. The code LS6 selected the lens that gave 6-point type, the code LS8 selected the lens that gave 8-point type, and so forth. The programs provided erroneous output if the number that was used in an LS code did not correspond to any point size that could be obtained with the lenses in the lens turret of the equipment that was used. The choice of the particular abbreviations, or "mnemonics," DL and LS was arbitrary, and it could have been changed in a very simple manner, as discussed in Section 7.4.

It should be mentioned that when several control codes were typed together, as in [IN, DL8, LS10], each code could also be typed within a separate pair of square brackets. The separating commas then were omitted. Thus [IN] [DL8] [LS10] could be typed instead of [IN, DL8, LS10]. Using separate square brackets for each control code was less convenient than using a single pair of square brackets and separating commas, but it is mentioned since some TYPRINT source material that has been prepared in the less convenient way is included in examples of the C.C.L. work. The order in which the DL and LS codes were typed was immaterial. Thus [IN, LS10, DL8] could be typed instead of [IN, DL8, LS10].

When the IN code was encountered by the computer program that dealt with TYPRINT input, the program automatically prepared to set material in a certain standard format. The characteristics of this format included the use of the type face on disc level 1, in 6-point type, a line length of 432 points and a page depth of 540 points. Any of the provisions of the IN code could be overruled by a later code in the input that applied to a feature of typography for which the convention of the standard format was not wanted. Thus in the coding [IN, DL6, LS8], the code DL6 overruled the selection of disc level 1 by the IN code, and the code LS8 overruled the selection of point size 6 by the IN code.

3.5 Setting Several Paragraphs

Section 3.4 describes the preparation of a text that consisted of a single paragraph for processing as TYPRINT input. In general, a text consists of several paragraphs. It is simpler, when using the Flexowriter, to indent a paragraph by pressing the tabulate key instead of using the space bar. To produce hard copy on the Flexowriter that was easy to proofread, when preparing TYPRINT input, the keyboard operator could leave blank lines between paragraphs, and could indent the first word of each paragraph by pressing the

```
[IN,DL8,LS8,NP]
     This example consists of several short paragraphs.
Each paragraph is indented in the input by use of the
tabulate key and in the output by the NP code which precedes
it in the input.   [NP]
     The first word of this paragraph is indented in the
manner which has just been explained.   A third paragraph
follows, to complete the example.[NP]
     This third paragraph completes the example.[EN]
```

This example consists of several short paragraphs. Each paragraph is indented in the input by use of the tabulate key and in the output by the NP code which precedes it in the input.

The first word of this paragraph is indented in the manner which has just been explained. A third paragraph follows, to complete the example.

This third paragraph completes the example.

Example 3.3. *The use of the NP code to start new paragraphs.*

```
[IN,DL6,LS8]
This sentence [NL]
is printed[NL]
on three lines.[EN]
```

This sentence
is printed
on three lines.

Example 3.4. *The use of the NL code to start new lines.*

tabulate key. The indention that was produced on the hard copy made it easy for the person reading proof to compare the hard copy with the manuscript. With the TYPRINT option of the PC6 programs, the computer ignored the tabulate key, and also ignored blank lines. In order that each paragraph beginning be indented in the typeset output, the computer had to be given an explicit instruction to indent at every paragraph break. For this purpose the code NP was typed between square brackets at the beginning of each paragraph, or at the end of the preceding paragraph for the second and later paragraphs in a text. It was important for the typist to remember that when the code NP was used, indention on the Flexowriter copy had to be produced by pressing the tabulate key. The space bar could not be used to indent the first word of a paragraph when the code NP was used, as this would have caused an error. No extra vertical space was left between paragraphs unless this was requested in the way that is described in Section 3.8.

The use of the NP code is illustrated by Example 3.3. The output was set in 8-point Highland type. The Flexowriter input was constructed in a very simple manner. First a left square bracket was typed to signal the beginning of a control code. The code IN was typed to show that a job was starting, and then a comma was typed because another control code followed. The code for Highland type, DL8, was found in Table 3.1, and this was typed, followed by a comma. The code LS8 was found in Table 3.3, typed to request the appropriate size of type, and followed by a comma. Then the code NP was typed to signal the beginning of a new paragraph. A right square bracket then was typed because text matter followed. The carriage return and tabulate key were pressed, and typing of the text was begun. The code NP was put between square brackets at the end of each paragraph until the end. A blank line was left between paragraphs for clarity in proofreading. The indention at the beginning of each new paragraph was obtained by pressing the tabulate key. The code EN was typed between square brackets at the end of the last paragraph. Once again the computer set the line length automatically to 432 points and the page length to 540 points, though the latter setting did not matter as the material did not fill a page.

3.6 Changing the Line Length and Starting a New Line

It was very easy to specify the line length, or "measure" in which material was to be set, when preparing TYPRINT input. A code was used that consisted of the letters LN followed by the number of

Photon points per line. This number is 71$\frac{1}{9}$ times the length of the line in inches. Thus the code LN360 was typed between square brackets to get a 360-point line. This is a little more than 5 inches long. The code LN504 was typed between square brackets to get a line that was 504 points long and so on. In the first example of this chapter, the initial coding [IN, DL6, LS8] was used to set 8-point Scotch type on a 432-point line. To set 8-point Scotch type in lines that were 360 points long, the coding [IN, DL6, LS8, LN360] was typed instead. The code LN360 overruled the selection of 432-point measure which the IN code provided, just as the DL6 and LS8 codes overruled the IN code's selection of type face and type size. Since the DL6, LS8 and LN360 codes overruled typographic details that were quite independent, their order did not matter, provided that they all came after the IN code. The coding [IN, LN360, LS8, DL6] or [IN, DL6, LN360, LS8] or [IN followed by any other permutation of DL6, LS8, and LN360, and then] would have had the same effect as [IN, DL6, LS8, LN360].

In general, a line break in the Flexowriter input did not cause a line break in the typeset product. The computer used words from successive lines of the Flexowriter input to fill a line of the Photon output. The NP code caused a new line to be started, with an indention, as explained in Section 3.5. Text matter could be made to start a new line without indention when it was set in type, by the inclusion of the code NL in square brackets in the input. A new line could be started on the Flexowriter after this NL code was typed, to facilitate proofreading. It was the NL code, however, and not the Flexowriter carriage return, that caused a new line to begin in the output at the appropriate position in the text. The use of NL code is illustrated by Example 3.4. The output would have had the same appearance if the input had been typed as follows:

```
[IN, DL6, LS8] This sentence [NL]
is printed [NL] on three lines. [EN]
```

The way in which the program dealt with spaces in the input requires further comment. The conventions that the PC6 programs used were to some extent arbitrary, but a set of conventions had to be introduced to deal with all the possibilities that could arise, and these conventions could be changed by methods that are discussed in Section 7.4. Each depression of the space bar on the Flexowriter punches a space code in the paper tape, and the computer had to have rules to deal with all the codes that it encountered in the input. The PC6 programs disregarded all spaces at the right of a line of input, between a nonblank character and an NL or NP code, and in lines that contained only spaces. The program treated spaces that started a line of input, or which occurred between other characters in a line of input, or which occurred between an NL or an NP code and the next non-blank character in the same line of input, as spaces which had to be preserved in the typeset output. Spaces which started a line of input, or which followed an NL or an NP code, produced output spaces which were not modified in the justification process.

3.7 Changing the Style Within a Text

In all the examples that are given in the preceding sections, a uniform type style is used throughout the texts that were set by the TYPRINT option. The user, however, could change type style within a text, and even within a line, as often as was necessary. In fact, each of the features of the format and typography could be changed independently of all others, so that, for example, the style of type could be changed independently of the size, or vice versa, or both could be changed together. Where a change was to take effect, the code for the changed appearance was inserted into the text. In Example 3.5, a sentence was set in 8-point Scotch type, with certain words italicized. The codes at the beginning of the Flexowriter copy started the setting in 8-point Scotch type. The DL14 code before each italicized word switched the type to Scotch Italic. The DL6 code after each italicized word switched the type style back to ordinary Scotch.

This method of changing styles within a text could be used to set headings and subheadings in a distinctive fashion. In Example 3.6, the first line was set in Century bold type, the second line was set in Scotch type, and the third line was set in Granite type. All three lines were set in 10-point size. In the Flexowriter input, the IN code showed that a job was starting, and incidentally specified the use of type on disc level 1 in 6-point size. These specifications were immediately overruled by the DL5 code, which requested Century bold type, and the LS10 code, which requested 10-point type. Since no further LS code was encountered in the input, all three lines were set under the control of the LS10 code, so that 10-point type was used throughout. The first line was set in Century bold type because of the DL5 code. The NL code then showed that a new line had to start, and the code DL6 requested Scotch type. The second line was set accordingly, and the codes NL and DL7 then showed that a new line had to start and that setting should continue in Granite type.

A different form of the same heading appears in Example 3.7. In the Flexowriter input, the first code IN showed that a job was start-

```
[IN,DL6,LS8]
The TYPRINT system consists of a [DL14]Flexowriter[DL6], a
[DL14]computer [DL6]and a [DL14]Photon [DL6]machine.[EN]
```

The TYPRINT system consists of a *Flexowriter*, a *computer* and a *Photon* machine.

```
[IN,DL5,LS10]
The TYPRINT System[NL,DL6]
Cooperative Computing Laboratory[NL,DL7]
Massachusetts Institute of Technology[EN]
```

The TYPRINT System
Cooperative Computing Laboratory
Massachusetts Institute of Technology

```
[IN,DL6,LS10]
The TYPRINT System[NL,LS9]
Cooperative Computing Laboratory[NL,DL8,LS8]
Massachusetts Institute of Technology[EN]
```

The TYPRINT System
Cooperative Computing Laboratory
Massachusetts Institute of Technology

Examples 3.5, 3.6, and 3.7. *Varying type face and size within the text.*

ing. The codes DL6 and LS10 caused the first line to be set in 10-point Scotch type. Then the code NL called for a new line of output, and the code LS9 caused this to be set in 9-point type. The DL6 code was still operative since no other DL code had been encountered, so the second line was set in Scotch type. Then the NL code called for a new line, and the DL8 and LS8 code caused the third line to be set in 8-point Highland type. Successive lines were set in the output as close together as possible in accordance with the vertical spacing convention that was implied by the IN code, i.e., that the distance between the base lines of two successive lines should be the point size of the type in the lower line.

3.8 Changing the Spacing Between Lines

The IN code caused the program to keep the space between successive lines of output to a minimum. Additional spacing could be obtained, however, by further coding, if it was needed. A small amount of extra spacing between lines of type is used in almost all printed matter, since most type faces were designed to accommodate it and, indeed, most require it for adequate readability. In the same way, most typewritten manuscripts are prepared with extra space between the lines, usually by double spacing.

Typically, the extra space used between lines of printed matter is a moderate fraction of the size of the type that is used. Thus 10-point type is normally used with 2 or 4 extra points of space between the lines. 14-point type is normally used with 4 or 6 points of extra space. The amount of extra space depends on the length of the line—the longer the line, the more spacing normally used. As an example of this increased spacing, consider 10-point type with 2 extra points between the lines. The type alone would occupy 10 points of space and so the type and extra spacing together occupy a total of 12 points of depth; the base line of a second line of type is 12 points below the base of the first line. Similarly, 10-point type with 4 extra points of space occupies a depth of 14 points.

To obtain such spacing in the output of the TYPRINT system, two codes were typed. The first code was LD2; this prepared the computer to receive the second code, which specified the actual distance which was to be left between the base lines of successive lines of type. For type which was to be set with the base lines 12 points apart, this code was BD12; for 14 points, BD14. The use of the BD code is illustrated by Examples 3.8, 3.9 and 3.10. To leave a certain number of inches between the base lines of successive lines of print, this number was multiplied by 71$\frac{1}{9}$, and written in a BD code. Thus the code BD71 left slightly less than 1 inch between successive base lines and the code BD36 left slightly more than $\frac{1}{2}$ inch between base lines. To provide double and triple spacing, the point size of the type was multiplied by 2 or 3 respectively and written in a BD code.

Sometimes extra space had to be added when a new line was started at a point in the text which was known when the input was typed on the Flexowriter. This happened when extra space was to be left be-

```
[IN,DL6,LS8,LD2,BD10,NP]
This paragraph is set by the TYPRINT system in 8 point Scotch type
with each line occupying a total of 10 points, leaving 2 points
of extra space between lines.   It is somewhat more readable than
the previous samples of 8 point Scotch type set without extra
space.[EN]
```

This paragraph is set by the TYPRINT system in 8 point Scotch type with each line occupying a total of 10 points, leaving 2 points of extra space between lines. It is somewhat more readable than the previous samples of 8 point Scotch type set without extra space.

```
[IN,DL6,LS8,LD2,BD14,NP]
This paragraph is set in the same 8 point Scotch type, but six extra
points are used between each line, thus there is a total of 14 points
occupied by each line.[EN]
```

This paragraph is set in the same 8 point Scotch type, but six extra points are used between each line, thus there is a total of 14 points occupied by each line.

```
[IN,DL6,LS12,LD2,BD14,NP]
This paragraph shows a larger size of Scotch (12 point) with 2 extra
points between each line.   As in the sample immediately above, each
line occupies a total of 14 points - hence the BD14 code is the
same for the two samples.[EN]
```

This paragraph shows a larger size of Scotch (12 point) with 2 extra points between each line. As in the sample immediately above, each line occupies a total of 14 points - hence the BD14 code is the same for the two samples.

Examples 3.8, 3.9, and 3.10. *Changing line spacing with the BD code.*

```
[IN,DL6,LS9,NP]
    Example of vertical spacing.   This sentence is set in 9
point type in the output.   Two blank lines, each of the height
of a line of 9 point type, are left after this paragraph in
the output, as a result of a VL code.   [NP,VL2,LS12]

    This paragraph is set in 12 point type.   Two blank lines,
each of the height of a line of 12 point type, are left after
this paragraph in the output, as a result of the VL code that
follows.   [NP,VL2,LS9]

    This paragraph is set in 9 point type to complete the example.[EN]
```

Example of vertical spacing. This sentence is set in 9 point type in the output. Two blank lines, each of the height of a line of 9 point type, are left after this paragraph in the output, as a result of a VL code.

This paragraph is set in 12 point type. Two blank lines, each of the height of a line of 12 point type, are left after this paragraph in the output, as a result of the VL code that follows.

This paragraph is set in 9 point type to complete the example.

Example 3.11. *Adding extra vertical space.*

```
[IN,DL6,LS8,LD2,BD12,NP]

     This example illustrates the effect of the VL code when
the BD code is in use.   In the printed output, the distance
between the base line of the letters in the last line of this
paragraph and the base line of the letters in the first line of
the next paragraph is twice the distance between the base lines
of letters in successive lines within each paragraph.   [NP,VL1]

     This sentence constitutes the second paragraph that is
needed to complete the example of the use of the BD code and
VL code in conjunction with each other.   [EN]
```

This example illustrates the effect of the **VL** code when the **BD** code is in use. In the printed output, the distance between the base line of the letters in the last line of this paragraph and the base line of the letters in the first line of the next paragraph is twice the distance between the base lines of letters in successive lines within each paragraph.

This sentence constitutes the second paragraph that is needed to complete the example of the use of the **BD** code and **VL** code in conjunction with each other.

Example 3.12. *The use of several codes to control vertical spacing.*

```
[IN,DL6,LS8]
Example of the VP code.   [NL,VP72]
Blank space of height one inch is left above this line.[EN]
```

Example of the VP code.

Blank space of height one inch is left above this line.

Example 3.13. *The VP code gave added flexibility in vertical spacing.*

```
[IN,DL6,LS8,AB24,NP]
     This example illustrates the use of the "added lead" codes.
Each paragraph is set in 8 point type, but an extra third of
an inch is left between paragraphs.   [NP,AL]
     This second paragraph continues the example.   A third
paragraph follows, to show that the AL code can be used
repeatedly.   [NP,AL]
     This third paragraph concludes the example, and shows that the
AB code need be given only once, even though several AL codes are
used subsequently.   [EN]
```

This example illustrates the use of the ‘added lead‘ codes. Each paragraph is set in 8 point type, but an extra third of an inch is left between paragraphs.

This second paragraph continues the example. A third paragraph follows, to show that the AL code can be used repeatedly.

This third paragraph concludes the example, and shows that the AB code need be given only once, even though several AL codes are used subsequently.

Example 3.14. *Control of vertical spacing with the AB and AL codes.*

```
[IN,DL6,LS8,LD2,BD10,AB24,NP]
     This example illustrates the use of the" added lead" codes.
Each paragraph is set in 8 point type, but an extra third of
an inch is left between paragraphs.   [NP,AL]
     This second paragraph continues the example.   A third
paragraph follows, to show that the AL code can be used
repeatedly.   [NP,AL]
     This third paragraph concludes the example, and shows that the
AB code need be given only once, even though several AL codes are
used subsequently.   [EN]
```

This example illustrates the use of the ‘added lead‘ codes. Each paragraph is set in 8 point type, but an extra third of an inch is left between paragraphs.

This second paragraph continues the example. A third paragraph follows, to show that the AL code can be used repeatedly.

This third paragraph concludes the example, and shows that the AB code need be given only once, even though several AL codes are used subsequently.

Example 3.15. *Another example of use of the AB and AL codes.*

tween paragraphs in the output. When the extra space consisted of some number of blank lines, it could be requested in TYPRINT input by typing a code that consisted of the letters VL followed by this number, all in brackets. Thus, to leave a blank line before a paragraph in the output, the codes [NP, VL1] were typed, at the end of the preceding paragraph in the input. The codes [NP, VL2] were typed in the input to provide two blank lines between paragraphs in the output. The codes [NP, VL3] were typed to leave three blank lines, and so forth. An NL code could be followed by a VL code to leave some number of blank lines before a nonindented line was started.

When a VL code was used, each blank line had the depth of the point size which was in force when the VL code was encountered. This is illustrated by Example 3.11. It was essential that the VL code follow the NP code in this example. If the codes [VL1, NP] had been typed at the end of a paragraph, the last line of the paragraph would have been double spaced below the last line but one, and no extra space would have been left between the paragraphs. This rule can be understood by reference to the way in which the computer program determined the space to be left between successive lines. When the computer was processing the input that was to be set on a line in the output, it did not finalize the space that was to be left above that line until the line was finished. The computer recognized the end of the last line of a paragraph by encountering an NP code. The code VL1 therefore applied to the next line, that is to the first line of the next paragraph, when [NP, VL1] was typed in the input. The first line of the next paragraph consequently would have the additional blank line left above it. However, when [VL1, NP] was typed, the VL1 code took effect before the NP code was encountered, and therefore applied to the last line of the paragraph. The code VL1 provided a blank line above the line to which it applied, so that [VL1, NP] put a blank line above the last line of the paragraph.

The VL code also could be used in conjunction with the BD code. When the spacing between successive lines of the printed output was specified by a BD code, the VL1 code left enough blank space for an extra line to be printed with the spacing that the BD code provided. A VL2 code left enough blank space for two such lines, and so forth. This is illustrated by Example 3.12.

Another code that was similar in its action to the VL code was used to space a single line by an additional amount that was measured in Photon points instead of in blank lines. The number of points was typed after the letters VP, all in brackets, at the appropriate position of the input. This is illustrated by Example 3.13.

Two further codes were used to increase the space above individual lines of output by the appropriate number of Photon points. This number was typed after the letters AB, all in brackets, at any position in the text before the place where it was to be used. The space was left in the output wherever the further code AL was typed between square brackets in the input. The computer provided extra space above the line of output which contained the material that followed the AL code in the input. The AL code could be used in as many lines as necessary without repeating the AB code. The AB and AL codes could be used to increase the space that was provided automatically when no other spacing codes were in use. The AB and AL codes also could be used to increase the space that was provided by a BD code.

The use of the AB and AL codes is illustrated by Examples 3.14 and 3.15. These codes together had the same effect as the VP code, but sometimes they were more convenient to use.

As mentioned earlier, whenever a BD code was used the code LD2 had to occur at some point before it in the input. The code LD1 was typed to suppress the effect of an LD2 code to make the spacing depend basically on the size of type that was in use.

3.9 Changing Margins and Vertical Alignment

All the examples of TYPRINT output that have been described so far started from the standard margin on the film that was determined by the position of a stop in the Photon machine (see Section 2.12). TYPRINT material was always set starting from this standard margin unless the position of a nonstandard margin was specified in the Flexowriter input by one code, and indention to this nonstandard margin was requested by another code. The position of the nonstandard margin was described by giving its distance from the standard margin, measured in Photon points. This distance was written after the letters LM, all in square brackets. Thus the code LM144 stated that the nonstandard margin was 144 Photon points (or roughly 2 inches) to the right of the standard margin. The LM code could be typed anywhere in the input before the point corresponding to which indention had to start in the output. The code MA was typed to start indention and the code NM was typed to restore the margin to its standard position. The codes MA and NM could be used repeatedly to start and to end several indented passages in a text which otherwise was not indented. The use of these codes is illustrated by Example 3.16. The position of the nonstandard margin could be changed by giving new LM codes. This is illustrated by Example 3.17.

```
[IN,DL6,LS8,LM144,NP]
     This example illustrates the use of indentation codes.
Following this sentence, the output is indented for two sentences as
a result of the code in the input that sets the position of the
non-standard margin, and the code that now follows in the input and
which requests indentation.  [MA,NL]
     This sentence is indented in the output as a consequence of the
     MA code.  It is indented in the input by use of the tabulate
     key which is ignored by the computer.[NM,NL].
This sentence starts at the standard margin in the output as a result
of the preceding NM code in the input.  The sentence that follows is
indented in the output as a result of another MA code in the input.
[MA,NL]
     This sentence is indented in the output as a result of the MA
     code preceding it.  Printing is started from the standard margin
     for the next sentence as a result of another NM code in the
     input.[NM,NL]
This sentence starts at the standard margin.[EN]
```

This example illustrates the use of indentation codes. Following this sentence, the output is indented for two sentences as a result of the code in the input that sets the position of the non-standard margin, and the code that now follows in the input and which requests indentation.

> This sentence is indented in the output as a consequence of the MA code. It is indented in the input by use of the tabulate key which is ignored by the computer.

This sentence starts at the standard margin in the output as a result of the preceding NM code in the input. The sentence that follows is indented in the output as a result of another MA code in the input.

> This sentence is indented in the output as a result of the MA code preceding it. Printing is started from the standard margin for the next sentence as a result of another NM code in the input.

This sentence starts at the standard margin.

Example 3.16. *Changing the left margin.*

```
[IN,DL6,LS8,NP]
     This paragraph starts from the standard margin.   The first
word is indented from the margin to mark the beginning of a paragraph.
The indentation in the input is provided by the tabulate key.   The
indentation in the output is produced by the NP code in the input.   The
line that follows this sentence is indented by 1 inch in the output as
a result of an LM code and an MA code.   [LM72,MA,NL]
     This line is indented by 1 inch in the output.   Another LM code
     and an MA code follow this sentence in the input, causing indentation
     of the output by another inch.[LM144,MA,NL]
          This sentence is indented by 2 inches from the standard margin
          in the output since the most recent LM code in the input
          contains 2 times 72.[LM36,NM,NL]
This sentence starts at the standard margin in the output as a consequence
of the preceding NM code in the input.   The LM code that precedes it is
used by the MA code that follows this sentence in the input, to indent
the next sentence by half an inch in the output.[MA,NL]
     This sentence is indented by half an inch in the output as just
     explained.[EN]
```

```
   This paragraph starts from the standard margin.  The first word is indented from the margin to mark the beginning of
a paragraph.  The indentation in the input is provided by the tabulate key.  The indentation in the output is produced by
the NP code in the input.  The line that follows this sentence is indented by 1 inch in the output as a result of an LM code
and an MA code.
                          This line is indented by 1 inch in the output.  Another LM code and an MA code follow this sentence
                          in the input, causing indentation of the output by another inch.
                                                    This sentence is indented by 2 inches from the standard margin in the output since
                                                    the most recent LM code in the input  contains 2 times 72.
This sentence starts at the standard margin in the output as a consequence of the preceding NM code in the input.  The
LM code that precedes it is used by the MA code that follows this sentence in the input, to indent the next sentence by
half an inch in the output.
             This sentence is indented by half an inch in the output as just explained.
```

Example 3.17. *Repeated changes of the left margin.*

When the line length was changed by an LN code, as described in Section 3.6, it was measured from the standard left margin even if a nonstandard left margin was in use. Thus the codes [LM72, MA, LN360] set type on a portion of the page that was 4 inches wide, since the line ended 5 inches from the standard margin as a result of the LN code and started 1 inch from this margin as a result of the LM and MA codes.

The IN code caused output to be justified automatically by the computer, except for the last line of a paragraph, and except for any line of output which was caused to end by an NL code in the input. It was possible to suppress the justification by typing the code NJ. When this code was typed, the same space was left in the output for each space that was included in the input, and the interword spaces were not altered to make the right margin straight. The computer still determined how many words were set on each line by reference to the line length and size of type that were in use. The computer set the maximum number of complete words on each line that was consistent with this line length and type size. The NL and NP codes still caused the start of a new line and a new paragraph. The effect of the NJ code is illustrated by Example 3.18. Justification could be restored after a nonjustified passage by typing the code JU.

Three further codes, FL, FC, and FR, were used to set material without altering the interword spaces to make both edges straight. The code FL had the same effect as the NJ code. The code FC centered each line between the margins that were in use. The code FJ moved each line of output to abut the right margin that was in use, leaving the left edge ragged. The use of these codes is illustrated elsewhere (in Examples 3.23 to 3.25).

When the same passage was set twice, first with justification and then without justification (by use of an NJ, FL, FC, or FR code) but all other features of the printing the same, it was possible for line breaks to occur at different positions in the justified and unjustified output that was produced by the PC6 programs. This was because a line was ended in unjustified material when the next word would cause overset, whereas this next word might be retained if justification were performed by "compression," as explained in Section 3.12.

3.10 Changing Page Length and Page Numbering

The IN code set the page length automatically to 540 Photon points, that is to about 7½ inches. To request a different page length, the length in inches was multiplied by 71⅑, and a code was typed that

```
[IN,DL3,LS8,NJ,NP]
     This example illustrates the redistribution of material
in lines when the TYPRINT
system is used with suppressed justification.
The line endings in the
input are irrelevant, just as when justification occurs.
A new line can be started in non-justified output by typing
an NL code in the input, just as for justified material.  [NL]
This sentence starts a new line in the output as a result of
the preceding NL code in the input.  [EN]
```

This example illustrates the redistribution of material in lines when the TYPRINT system is used with suppressed justification. The line endings in the input are irrelevant, just as when justification occurs. A new line can be started in non-justified output by typing an NL code in the input, just as for justified material.
This sentence starts a new line in the output as a result of the preceding NL code in the input.

Example 3.18. *Suppressed justification of lines.*

```
[IN,(plus=CH1),(lsb=DL9,CH68),(rsb=DL9,CH81),DL6,LS10,FC,LD2,BD15]

To photocompose the explanation of this example it is necessary
to type as follows:[NL] The equation 7[plus]4=11 is set by
typing 7[lsb,DL6]plus[rsb,DL6]4=11.   [EN]
```

To photocompose the explanation of this example it is necessary to type as follows:
The equation 7 + 4 = 11 is set by typing 7[plus]4 = 11.

Example 3.19. *The definition and use of some special characters.*

consisted of the letters PD followed by the result, all in brackets. Thus the code [PD711] produced a 10-inch page. The user could request that a new page start at a particular position in the typeset output by typing the code [PG] at the corresponding position of the input.

The output of the Photon machine consists of a roll of film or sensitized paper, as described in Chapter 2. The PC6 programs automatically left a quarter-inch of blank film between the type that was to appear on the successive pages into which the continuous film was cut. This blank film was left to facilitate the cutting and developing in the darkroom. Sometimes it is necessary to produce the output in a continuous manner, without any subdivision into material for successive pages. To do this, the code [KP] was typed in the input.

It is also necessary at times to ensure that a page break does not occur within some portion of a text. For example, in setting a bibliography there are certain basic fragments of information that must not be broken between pages. A fragment of this type may be continued below the usual base line of a page to avoid breaking it. The codes CP and AP were typed in the input respectively at the beginning and end of each such fragment. If the material that was typed between an occurrence of the code CP and the next occurrence of the code AP caused output to be set below the usual end of the page, then the material that followed the code AP started a new page automatically in the output.

The PC6 programs provided a limited page numbering facility for TYPRINT input. The codes PN1, PN2, and PN3 caused page numbers to be set at the left, center, and right, at the foot of the page. The point size and disc level to be used for the page numbers were specified by codes that were analogous to the LS and DL codes, but which used the letters LP and DP. Thus, the codes [PN2, LP8, DP16] centered the page numbers at the foot of the page, in 8-point type taken from disc level 16. The page numbering always started at 1. Many typesetting jobs require facilities for starting page numbers at some other specified value, for alternating left and right placement of the number, and for putting it at the head of the page.

3.11 Setting Special Characters

Several punctuation marks and common symbols that are on the standard Flexowriter keyboard could be included in TYPRINT input and the program provided the Photon codes for these characters in the output, just as it dealt with letters and digits. The characters which could be typed in this way were as follows:

, . : - / = * $? ( )

The characters " and ' on the Flexowriter produced the characters “ and ’, respectively, in the typeset output. Any character on the Photon disc could be typeset in the output by typing the name of the character, in square brackets, at the point in the text at which it had to appear. Thus, in Example 3.19 the names “plus,” “lsb,” and “rsb” were used, respectively, for the symbols +, [, and]. The user could employ any name that seemed convenient for a special character, provided that it consisted of 6 or fewer lower-case letters. The name had to be defined before the first occasion on which it was used in a TYPRINT run, by equating it to a code or pair of codes which specified the position on the Photon disc of the special character for which it stood. A special character definition was in effect a control code. It was typed between square brackets, by itself or with other control codes from which it was separated by commas. It consisted of a left parenthesis (, then the special character name, then an equality symbol = , and then the code or codes which specified the character's position on the disc, and then a right parenthesis) . Two codes usually were needed: the DL code which specified the disc level of the character, and a code which consisted of the letters CH followed by the sequence number of the character within that disc level. Thus the left square bracket on disc 78 is on disc level 9, with sequence number 68, so the name lsb was defined to stand for the left square bracket by typing

(1sb = DL9, CH68)

The disc level and sequence number codes were separated by a comma when they both occurred in a special character definition.

Some symbols that are used frequently may appear on every level of a Photon disc. Thus the + symbol appears on every level of disc 78 and it has sequence number 1 in every instance. The special character definition then required only that the sequence number be specified, as in the definition (plus = CH1). When a special character name was encountered in TYPRINT input after it had been defined, the program provided codes to move the disc to the level which was specified in the special character definition. The annotator had to provide a DL code after the special character to restore the disc level to that which was previously in use, if this differed from the level which contained the special character. This was not necessary if the special character definition did not contain a DL code. The DL code could be omitted from a special character definition when the character appeared on just one level of the disc, if it was used throughout a TYPRINT run in contexts which were set from that same disc level. The special characters on a Photon disc are listed on charts that the manufacturer provides. Discs that carry extensive selections of mathematical and other scientific symbols are used by the American Mathematical Society and the American Chemical Society in Photon 200 machine installations.

3.12 Horizontal Spacing Control

With certain exceptions, each space that was typed on the Flexowriter produced a space in the phototypeset output. The exceptions were spaces that followed a word set at the end of a line of output, provided that this did not correspond to a space in the input that was separated from the word by a carriage return. Spaces that followed a carriage return in the input were retained in the output, and were left at the beginning of an output line or between two of its words depending on whether the preceding word ended or did not end a line of the output. A code of the form HS*n*, where *n* stood for an integer, had the same effect as pressing the space bar *n* times. The computer provided the correct number of spaces after the last nonblank character of each line of input, that is, two spaces for a period and one space for all other characters, and these were treated just as if they had been typed.

An output space was produced from an input space in two stages. The program first provided for the production of a space that was 6 relative units wide in the output. This is the width of a letter "t" in most styles of type. In the second stage, spaces that occurred between words in the output were expanded or contracted by the computer as justification required. Spaces that preceded the first nonblank character in a line of output were not altered in this process.

It was possible to set lower and upper limits on the amount of space that could be left between successive words as a result of the justification process. The minimum interword space expressed in relative units was typed after the letters LO. The maximum interword space was set in the corresponding manner by typing a code with the letters HI instead of LO. The use of these codes requires some further explanation of the way in which the program performed the justification. The program used a table that contained the relative widths of all the letters of the alphabet, and of all the other symbols which were on the Photon disc. As the program processed the input material, it kept a record of the total width of the material on the line of output which it was preparing. When the program encountered a letter that made this amount exceed the required page width, the program completed the word and decreased the distance that was left between successive words in the line, so that the typeset line ended at the appropriate margin in the output. The program then checked that the distance between successive words had not been reduced below the allowed minimum interword space. Unless the user had specified this minimum, by use of an LO code, the program automatically set it to 4 relative units, that is, the width of a letter "i" in most styles of type. If the space between words was reduced below the allowed minimum, the program removed the last word from the line which had just been formed, leaving it to be put in the subsequent line of output. The program then expanded the spaces between all the words that remained, to ensure that the last of these would end at the appropriate margin in the output. The program tested to see if the space that was then provided between successive words exceeded the allowed maximum. In practice, this maximum was set automatically to a very large value, and the HI code was not used. It was hoped that at a later juncture the HI code would be used and that hyphenation would be performed by the computer when the allowed maximum was exceeded. It must be stressed that this goal was not reached in the C.C.L. typesetting project at M.I.T.

3.13 Tabular Material

Three further codes were used in the input to the TYPRINT system to set tabular material. Each line of output could consist of portions that were set between pairs of margins whose positions were specified in the input by "section" codes. Twenty or fewer pairs of margins could be specified in this way. The positions of each pair of margins needed to be specified only once, no matter how many times they were used in successive lines of the output. The pairs of margins were numbered 1 to 20. Whenever a portion of the input text was to be set between a pair of margins, a code was given that indicated, by reference to this numbering, which pair was to be used. Material could be left-adjusted, centered, or right-adjusted between any pair of margins, regardless of how the material that was set between the other pairs of margins was treated. Each pair of margins constituted a "section" in the terminology of the PC6 programs. The positions of the section margins were measured in Photon points from the standard margin (see Section 3.9). A control code had to be typed, to specify the positions of a section's margins, at some point in the input text before the first material that was to appear in that section in the output. This code consisted of a left parenthesis, then the letter S, and then the number that had been given to the section. This could be any number from 1 to 20 that had not been given to another section. It was followed by an = symbol. The letters LM were typed next, followed by the distance of the left margin from the standard margin of the output, measured in Photon points. A comma was typed next, then the letters RM, and then the distance of the right margin from the standard left margin of the output, also measured in Photon points. Finally, a right parenthesis was typed to indicate the end of the code. Thus the code (S7=LM144, RM216) defined the left and right margins of section 7 to be approximately 2 and 3 inches from the standard margin of the output. It was usual to number the sections consecutively from left to right for the convenience of the user, but this was not necessary for the program.

The TYPRINT option put material in the output in the order in which it was encountered in the input. The output was produced line by line, going from left to right in each line. To set material with section 1 margins, the code XS1 was typed in the input immediately before the first character to go into this section. To set material with section 2 margins, the code XS2 was typed and so on. Thus if section 7 had been defined by the codes (S7=LM144, RM216), then after an XS7 code was encountered the input material was set between margins that were 144 points and 216 points from the standard margin, until an XS code that referred to another section occurred in the input. If the material that followed the XS7 code required more than 72 points of type in the output, then it was set automatically on more than one line. The material that followed this in the input then appeared in the output on a line and to the right of the last of the characters that were set in section 7, or somewhere lower on the output page. This was done so that items encountered consecutively in the input could be set in successive sections of one line of output, as in a table with several entries on each line. The IN code had the effect of (S1=LM0, RM432), XS1 in addition to its other functions, and the NL, NP, and PG codes also had the effect of an XS1 code in addition to starting new lines, paragraphs, and pages. If an XS code was encountered that referred to a section which was on the left of the section in current use, a new line was started automatically. Increasing the section number by 20 (or 40) in an XS code set material in a new line (or page) of the section in question without the need for an NL or PG code.

The use of the section codes is illustrated by the production of a table of English monarchs of the 11th and 12th centuries. In Example 3.20, a single size and style of type is used. The tabular appearance of the Flexowriter copy was obtained by use of the tabulate key and was obtained solely for the convenience of the user. It was ignored by the computer, which depended on the XS codes to produce tabulation in the output. No blanks could occur in this input anywhere between an XS code and the first character which followed that code and which was to be printed in the output. If any blanks had occurred in this position, they would have been converted to spaces in the output that followed the left margin of the section. This would have spoiled the tabular effect. Occasionally it was appropriate to leave blanks before the first character, when several lines were to be set in a section and the first of these was to be indented with respect to the section's left margin. This was rather exceptional however. It was possible to set column headings in different sizes and/or styles of type than the entries in a table. This is illustrated by Example 3.21. In Example 3.22, the three columns are set in Century bold, Scotch, and Scotch Italic, respectively.

The tabular appearance of the Flexowriter copy was produced by use of the tabulate key for the convenience of the user as explained above. The XS codes were responsible for the tabular appearance of the output. The first character after the XS code that was not part of another code was set after the appropriate margin in the output. Thus the "M" of "Monarch" was aligned with the "W" of "William" in the output, even though they were not aligned in the input. The factor that determined the alignment in the output was the occurrence of the "M" and "W" as the first text characters after the XS3 codes.

In Example 3.23, the first column was centered and the second column was right adjusted. The heading and the second column were set in Highland type, the dynasty names in Scotch, and the dates in Granite.

Alignment could be obtained within a column of output by treating it as two sections, of which the first was right-adjusted and the second was left-adjusted. This was useful in aligning a column of numbers on a decimal point. It is illustrated by the setting of the column of monarchs aligned by the ends of their names in Example 3.24.

```
[IN][DL6][LS8][(S1=LM72,RM180)][(S2=LM180,RM288)][(S3=LM300,RM360)]

[XS21]NORMANS           [XS2]William I           [XS3]1066
                        [XS22]William II         [XS3]1087
                        [XS22]Henry I            [XS3]1100
                        [XS22]Stephen            [XS3]1135
[XS21]PLANTAGENETS      [XS2]Henry II            [XS3]1154
                        [XS22]Richard I          [XS3]1189
                        [XS22]John               [XS3]1199           [EN]
```

NORMANS	William I	1066
	William II	1087
	Henry I	1100
	Stephen	1135
PLANTAGENETS	Henry II	1154
	Richard I	1189
	John	1199

Example 3.20. *The use of section codes to produce a table.*

```
[IN]
[(S1=LM72,RM180)][(S2=LM180,RM288)][(S3=LM300,RM360)]
[DL8][LS10]
[XS21]Dynasty              [XS2]Monarch        [XS3]Accession
[DL6][LS8][VL2]
[XS1]NORMANS               [XS2]William I      [XS3]1066
                           [XS22]William II    [XS3]1087
                           [XS22]Henry I       [XS3]1100
                           [XS22]Stephen       [XS3]1135
[XS21]PLANTAGENETS         [XS2]Henry II       [XS3]1154
                           [XS22]Richard I     [XS3]1189
                           [XS22]John          [XS3]1199     [EN]
```

Dynasty	Monarch	Accession
NORMANS	William I	1066
	William II	1087
	Henry I	1100
	Stephen	1135
PLANTAGENETS	Henry II	1154
	Richard I	1189
	John	1199

Example 3.21. *Using a different type size and style for column headings.*

```
[IN]
[(S1=LM72,RM180)][(S2=LM180,RM288)][(S3=LM300,RM360)]
[DL5][LS10]
[XS21]Dynasty                 [DL6][XS2]Monarch          [DL14][XS3]Accession
[LS8][VL2]
[DL5][XS21]NORMANS            [DL6][XS2]William I        [DL14][XS3]1066
                              [DL6][XS22]William II      [DL14][XS3]1087
                              [DL6][XS22]Henry I         [DL14][XS3]1100
                              [DL6][XS22]Stephen         [DL14][XS3]1135
[DL5][XS21]PLANTAGENETS       [DL6][XS2]Henry II         [DL14][XS3]1154
                              [DL6][XS22]Richard I       [DL14][XS3]1189
                              [DL6][XS22]John            [DL14][XS3]1199 [EN]
```

```
Dynasty             Monarch             Accession


NORMANS             William I           1066
                    William II          1087
                    Henry I             1100
                    Stephen             1135
PLANTAGENETS        Henry II            1154
                    Richard I           1189
                    John                1199
```

Example 3.22. *Varying type style by column.*

```
[IN]
[(S1=LM72,RM180)][(S2=LM180,RM288)][(S3=LM300,RM360)]
[DL8][LS10]
[XS21][FC]Dynasty                [XS2][FR]Monarch            [XS3][FL]Accession
[LS8][VL2]
[XS21][DL6][FC]NORMANS           [XS2][DL8][FR]William I     [XS3][DL7][FL]1066
                                 [XS22][DL8][FR]William II   [XS3][DL7][FL]1087
                                 [XS22][DL8][FR]Henry I      [XS3][DL7][FL]1100
                                 [XS22][DL8][FR]Stephen      [XS3][DL7][FL]1135
[XS21][DL6][FC]PLANTAGENETS      [XS2][DL8][FR]Henry II      [XS3][DL7][FL]1154
                                 [XS22][DL8][FR]Richard I    [XS3][DL7][FL]1189
                                 [XS22][DL8][FR]John         [XS3][DL7][FL]1199
```

```
        Dynasty                    Monarch   Accession

        NORMANS                  William I   1066
                                William II   1087
                                   Henry I   1100
                                   Stephen   1135
      PLANTAGENETS                Henry II   1154
                                 Richard I   1189
                                      John   1199
```

Example 3.23. *Setting centered, left-adjusted, and right-adjusted columns. (Note use of FC, FR, and FL codes. See page 50.)*

```
[IN][DL6][LS10]
[(S4=LM180,RM270)][(S5=LM276,RM288)]
[XS24][FR]William         [XS5][FL]I
[XS24][FR]William         [XS5][FL]II
[XS24][FR]Henry           [XS5][FL]I
[XS24][FR]Stephen
[XS24][FR]Henry           [XS5][FL]II
[XS24][FR]Richard         [XS5][FL]I
[XS24][FR]John                                   [EN]
```

```
William  I
William  II
  Henry  I
Stephen
  Henry  II
Richard  I
   John
```

Example 3.24. *Internal alignment of a single column, using FR and FL codes.*

```
[IN]
[(S1=LM72,RM180)][(S2=LM180,RM288)][(S3=LM300,RM360)]
[(M1=DL6,FC)(M2=DL8,FR)(M3=DL7,FL)]
[DL8][LS10]
[XS21][FC]Dynasty               [XS2][FR]Monarch          [XS3][FL]Accession
[LS8][VL2]
[XS21][XM1]NORMANS              [XS2][XM2]William I       [XS3][XM3]1066
                                [XS22][XM2]William II     [XS3][XM3]1087
                                [XS22][XM2]Henry I        [XS3][XM3]1100
                                [XS22][XM2]Stephen        [XS3][XM3]1135
[XS21][XM1]PLANTAGENETS         [XS2][XM2]Henry II        [XS3][XM3]1154
                                [XS22][XM2]Richard I      [XS3][XM3]1189
                                [XS22][XM2]John           [XS3][XM3]1199

                                                          [EN]
```

```
        Dynasty                   Monarch   Accession

        NORMANS                  William I    1066
                                William II    1087
                                   Henry I    1100
                                   Stephen    1135
      PLANTAGENETS                Henry II    1154
                                 Richard I    1189
                                      John    1199
```

Example 3.25. *Abbreviation of codes by using macros. Compare with Example 3.23.*

3.14 Composite Codes (Macros)

The discussion of tabular material in the preceding section provides a convenient introduction to the use of "macros," or abbreviations for combinations of codes that were used repetitively. Example 3.23 used the combination of the two codes DL6 and FC twice, the combination of the codes DL8 and FR seven times, and the combination of the codes DL7 and FL seven times. Each of these combinations of codes could be abbreviated as a macro. In Example 3.25, macros were used in the input to produce the same output as that of Example 3.23. In this example, the abbreviations, or macros, were the codes M1, M2, and M3. The letter M followed by any number between 1 and 25 could be used as a macro in TYPRINT input. The number that was used was quite arbitrary otherwise. Thus M17, M11, and M25 could be used instead of M1, M2, and M3 in Example 3.25.

Each macro had to be defined at some point in the text before it was first used, by typing a left parenthesis, then the letter M and the number that had been chosen for the macro, then an = symbol, then the codes which the macro abbreviated, and then a right parenthesis at the end. Any number of codes could appear in this way in the definition of a macro. To execute a macro, a code was typed that consisted of the letters XM followed by the macro number.

3.15 Processing TYPRINT Input

After the TYPRINT control codes had been written into a manuscript or typescript, the paper tape and hard copy were prepared by standard keyboard operation of the Flexowriter. Typing mistakes that were noticed as they were made could be corrected by manually moving back the tape, and then punching the delete code (holes in all channels) into the erroneous row or rows of tape by pressing the "delete" key. This is standard Flexowriter practice. The program that dealt with TYPRINT input also permitted the keyboard operator to press the backspace key once or several times and then to type correct characters over the incorrect matter on the hard copy. Each backspace code advanced the tape although it moved the carriage backward. The program dealt with the backspace codes by deleting each sequence of backspace codes and the sequence of characters of equal length that preceded it.

The hard copy that was typed on the Flexowriter could be proofread for typographic errors. If any were found, the conventional Flexowriter tape correction procedure could be used to produce a correct tape. In this procedure, the tape that contained errors was placed in the reading unit of the Flexowriter. Its contents were punched automatically into a new tape and typed on the stationery in the carriage, until the operator stopped the process by pressing a key. The operator did this whenever material that was in error in the first tape was about to be typed. On these occasions, the operator typed the correct material, allowed the first tape to advance until the errors had passed the reading head without being copied into the second tape, and then pressed a key to resume the automatic copying and typing process.

Corrections of a limited nature could also be made by the editing feature of the TYPRINT option. This was rather weak, as relatively little effort was expended on it. It had some use, however, and is adequate as an example of a mechanized editing process. Possible improvements are discussed later.

3.16 Computer Editing

One or more lines of a text that had been typed on a Flexowriter could be altered by typing "editing commands" that specified the changes, also on a Flexowriter, and providing the paper tape of the original text and the paper tape punched with the editing commands as input to a TYPRINT run. Each editing command started with the capitalized word LINE, followed by the enumeration of the line to which it referred, counted within the hard copy for the entire job. A comma was typed after this number, and then a capitalized verb that specified the nature of the correction. The word ERASE was used to delete the line completely, so that to delete an entire line of the hard copy, the edit instruction was of the form

LINE n, ERASE

where n stands for the line number of the line to be erased. Part of a line was deleted, and the blank space that would have been left was closed out, by an instruction of the form

LINE n, DELETE ". . . . ."

where the portion to be deleted was given between quote marks. The letter n again stands for the line number. This method of identifying the portions to be deleted was not sufficient, for example, if one occurrence of a word that appeared several times on the line was to be deleted. The particular occurrence that was to be deleted then could be identified by giving the word (or if necessary, words) that preceded it, in an instruction of the form

LINE *n*, DELETE ". . . ." AFTER ". . . . ."

Alternatively, the portion that was to be deleted could be identified by giving the word (or if necessary words) that followed it, in an instruction of the form

LINE *n*, DELETE ". . . ." BEFORE ". . . ."

To insert material in a given line, the word (or words) that the insertion was to follow could be specified in an instruction of the form

LINE *n*, INSERT ". . . ." AFTER ". . . . ."

Alternatively, the word (or words) that the insertion was to precede could be specified in an instruction of the form

LINE *n*, INSERT ". . . ." BEFORE ". . . . ."

To replace a portion (or all) of a line with some new material, an instruction was used that was of the form

LINE *n*, REPLACE ". . . ." WITH ". . . . ."

When a text was edited in this way, the program could be controlled very easily to provide either (1) a paper tape which was punched with the edited text in Flexowriter code and which operated the Flexowriter to type the edited text mechanically, and/or (2) a paper tape that was punched with the edited text translated into Photon code. The program's operation could be arrested when Flexowriter tape was produced, without entering the translation stage that formed the Photon paper tape, and it was expedient to do this if there were so many corrections that the edited text required proofreading. On the other hand, it was expedient to suppress the production of the Flexowriter tape and to proceed automatically to the translation if there were few corrections and the user was confident that these had been specified correctly.

3.17 Run Processing

In principle, the paper tape that was punched on the Flexowriter from an annotated manuscript could be given to the computer operating staff of an IBM 709/90 installation, perhaps with an editing tape, and a Photon tape returned in due course by the computer staff without the user having to take any note of the computer run. All the users who experimented with the PC6 programs were very interested in the computer phase, and in fact it did affect them if the input was in error and mistakes had to be located.

The computer stage consisted of input conversion, computer processing, and output conversion. The input conversion stage consisted of the production of binary cards punched with Flexowriter tape codes on the modified IBM 047 punch, as described in Section 2.3. A "text deck" was formed from the text tape, and an "editing deck" from the tape that was punched with the editing commands. The operating staff formed an input deck from (1) a standard TYPRINT control deck, (2) the text deck, and (3) the editing control cards and editing deck. The input deck was batched with other monitor jobs to be written on magnetic tape off-line.

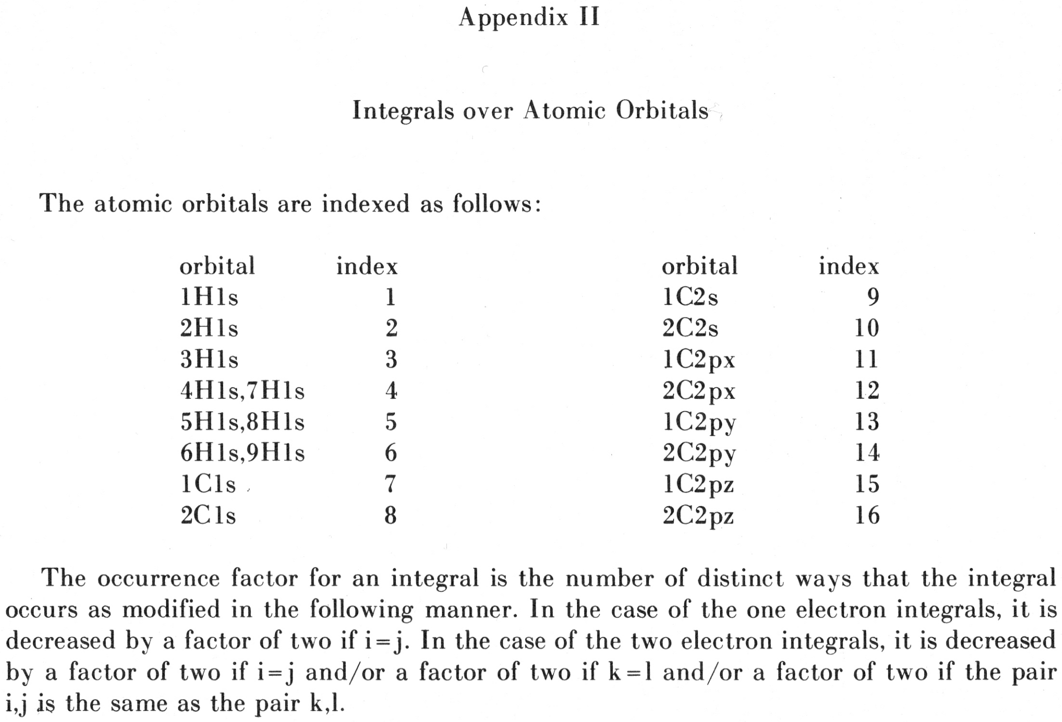
Computer runs with the PC6 programs could fail, just as any computer run may fail. The program could be run in either of two modes by setting a switch on the console up for one mode and down for the other mode. In one mode, the computer recorded a considerable amount of information concerning the progress of a run on magnetic tape while the run was in progress. This slowed down the operation of the computer, but it helped the staff to find the source of trouble if the run was unsuccessful. In the second mode of operation, the detailed history of the run was not kept, but some brief comment was recorded at the end of each of a few major stages of the job. The location of input errors in jobs that failed is discussed in Section 4.7. Job failures were due mainly to incorrect control codes and to incorrect editing commands.

Output conversion consisted of the preparation of Photon paper tape from the magnetic tape output of the computer run. This has been described in Section 2.5.

3.18 Film Preparation

The Photon tape punched from the output of successful TYPRINT runs was used to operate the Photon film setting equipment. It was customary in the C.C.L. to set about four pages continuously and then to develop these while setting continued with another magazine in use, as described in Section 2.11. The film might contain errors as a result of incorrect input, or machine errors in the equipment that was used in some stage of the work. Depending on the extent of the error, jobs were rerun *in toto* or in part, and if necessary, pages of film corrected by "stripping in," that is, by cutting and pasting.

Some further samples of material set with TYPRINT are shown in Examples 3.26 through 3.30.



Appendix II

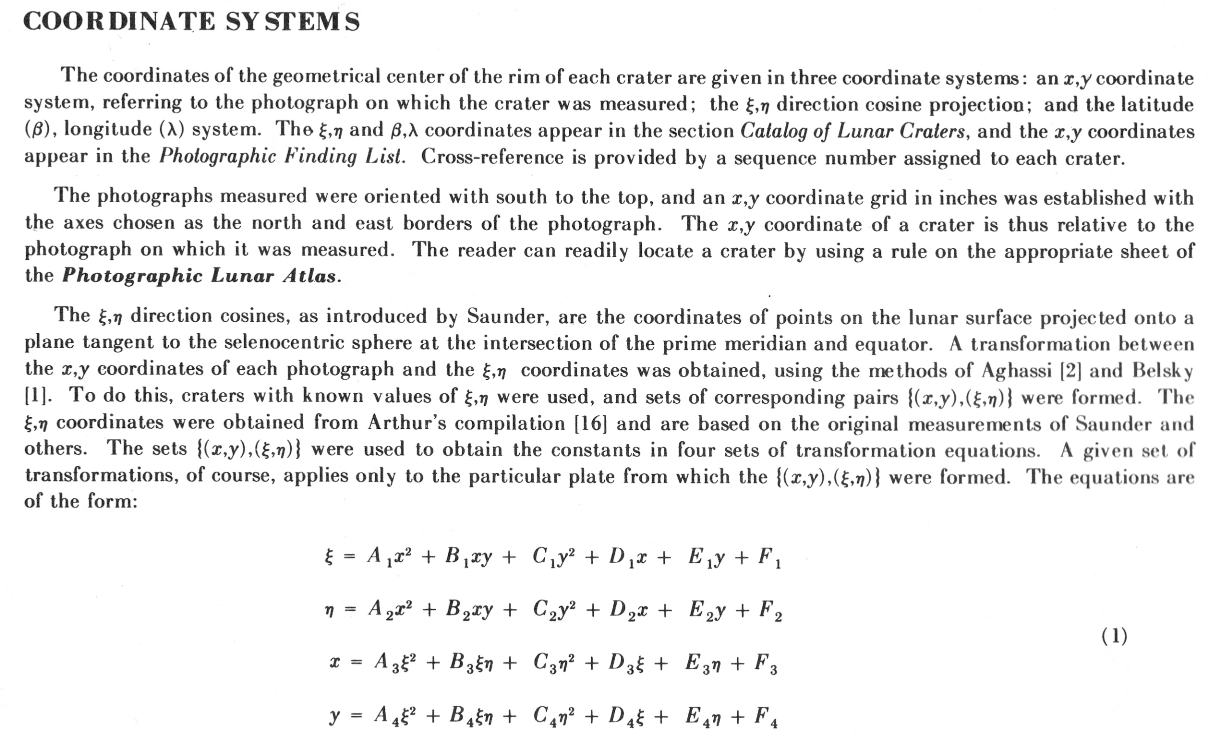
Integrals over Atomic Orbitals

The atomic orbitals are indexed as follows:

orbital	index	orbital	index
1H1s	1	1C2s	9
2H1s	2	2C2s	10
3H1s	3	1C2px	11
4H1s,7H1s	4	2C2px	12
5H1s,8H1s	5	1C2py	13
6H1s,9H1s	6	2C2py	14
1C1s	7	1C2pz	15
2C1s	8	2C2pz	16

The occurrence factor for an integral is the number of distinct ways that the integral occurs as modified in the following manner. In the case of the one electron integrals, it is decreased by a factor of two if i=j. In the case of the two electron integrals, it is decreased by a factor of two if i=j and/or a factor of two if k=l and/or a factor of two if the pair i,j is the same as the pair k,l.

Example 3.26. *Sample of TYPRINT output: Ph.D. dissertation of R. M. Pitzer. (PC6 project 3. See Section 1.9.)*



COORDINATE SYSTEMS

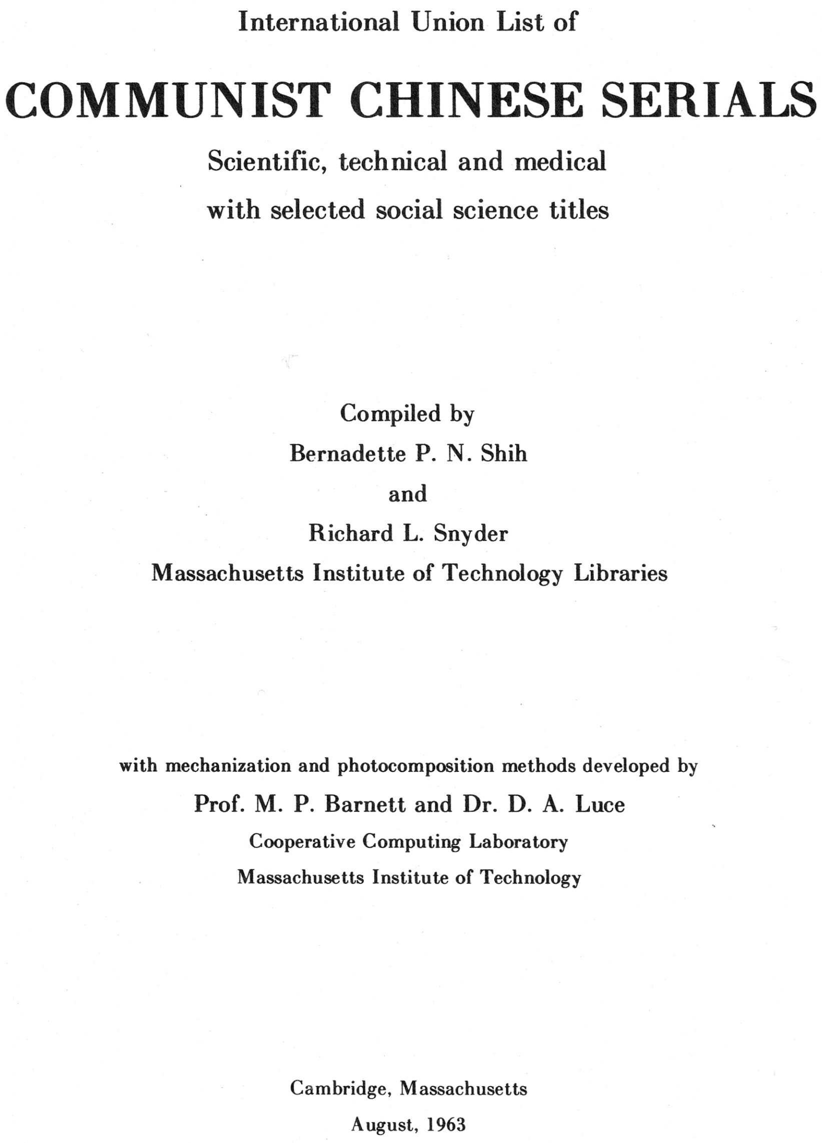
The coordinates of the geometrical center of the rim of each crater are given in three coordinate systems: an x,y coordinate system, referring to the photograph on which the crater was measured; the ξ,η direction cosine projection; and the latitude (β), longitude (λ) system. The ξ,η and β,λ coordinates appear in the section *Catalog of Lunar Craters*, and the x,y coordinates appear in the *Photographic Finding List*. Cross-reference is provided by a sequence number assigned to each crater.

The photographs measured were oriented with south to the top, and an x,y coordinate grid in inches was established with the axes chosen as the north and east borders of the photograph. The x,y coordinate of a crater is thus relative to the photograph on which it was measured. The reader can readily locate a crater by using a rule on the appropriate sheet of the ***Photographic Lunar Atlas***.

The ξ,η direction cosines, as introduced by Saunder, are the coordinates of points on the lunar surface projected onto a plane tangent to the selenocentric sphere at the intersection of the prime meridian and equator. A transformation between the x,y coordinates of each photograph and the ξ,η coordinates was obtained, using the methods of Aghassi [2] and Belsky [1]. To do this, craters with known values of ξ,η were used, and sets of corresponding pairs $\{(x,y),(\xi,\eta)\}$ were formed. The ξ,η coordinates were obtained from Arthur's compilation [16] and are based on the original measurements of Saunder and others. The sets $\{(x,y),(\xi,\eta)\}$ were used to obtain the constants in four sets of transformation equations. A given set of transformations, of course, applies only to the particular plate from which the $\{(x,y),(\xi,\eta)\}$ were formed. The equations are of the form:

$$\begin{aligned}
\xi &= A_1x^2 + B_1xy + C_1y^2 + D_1x + E_1y + F_1 \\
\eta &= A_2x^2 + B_2xy + C_2y^2 + D_2x + E_2y + F_2 \\
x &= A_3\xi^2 + B_3\xi\eta + C_3\eta^2 + D_3\xi + E_3\eta + F_3 \\
y &= A_4\xi^2 + B_4\xi\eta + C_4\eta^2 + D_4\xi + E_4\eta + F_4
\end{aligned} \tag{1}$$

Example 3.27. *Sample of TYPRINT output: portion of the* Catalog of Lunar Craters. *(PC6 project 9. See Section 1.9.)*



International Union List of

COMMUNIST CHINESE SERIALS

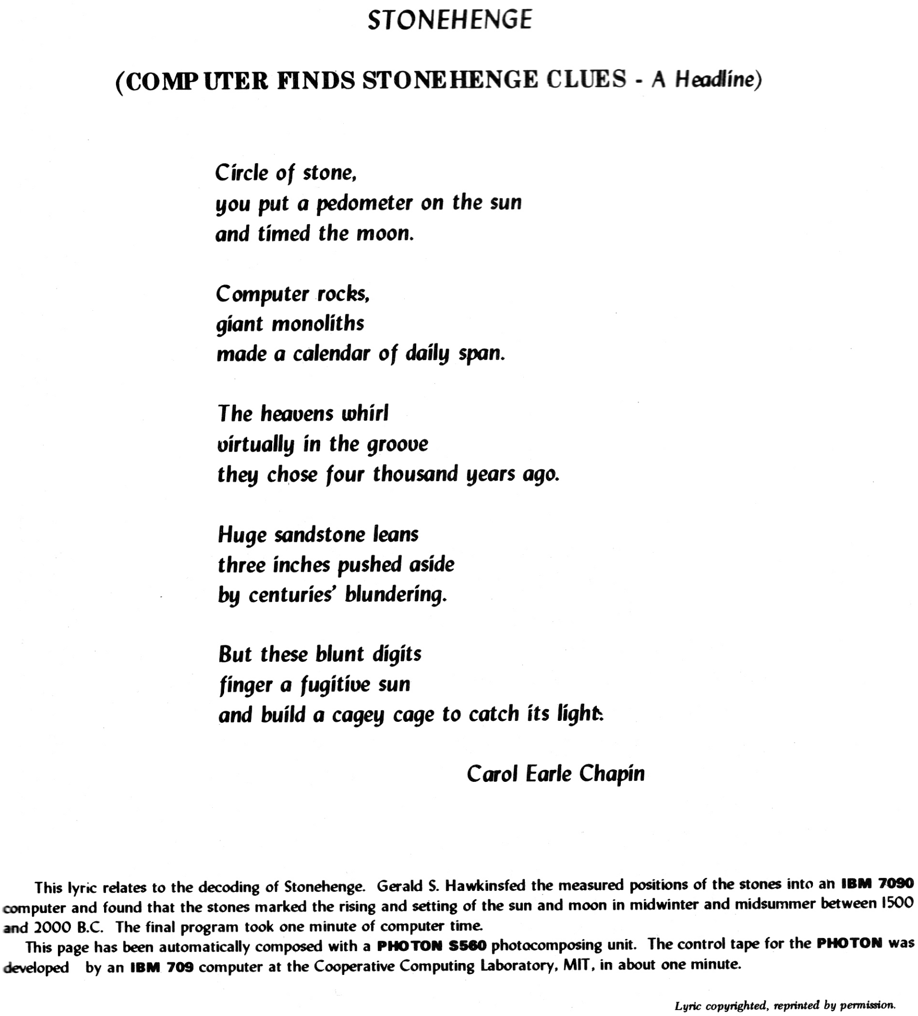
Scientific, technical and medical
with selected social science titles

Compiled by
Bernadette P. N. Shih
and
Richard L. Snyder
Massachusetts Institute of Technology Libraries

with mechanization and photocomposition methods developed by
Prof. M. P. Barnett and Dr. D. A. Luce
Cooperative Computing Laboratory
Massachusetts Institute of Technology

Cambridge, Massachusetts
August, 1963

Example 3.28. *Sample of TYPRINT output: title page of* Union List. *(PC6 project 4. See Section 1.9.)*



STONEHENGE

(COMPUTER FINDS STONEHENGE CLUES - A *Headline*)

Circle of stone,
you put a pedometer on the sun
and timed the moon.

Computer rocks,
giant monoliths
made a calendar of daily span.

The heavens whirl
virtually in the groove
they chose four thousand years ago.

Huge sandstone leans
three inches pushed aside
by centuries' blundering.

But these blunt digits
finger a fugitive sun
and build a cagey cage to catch its light.

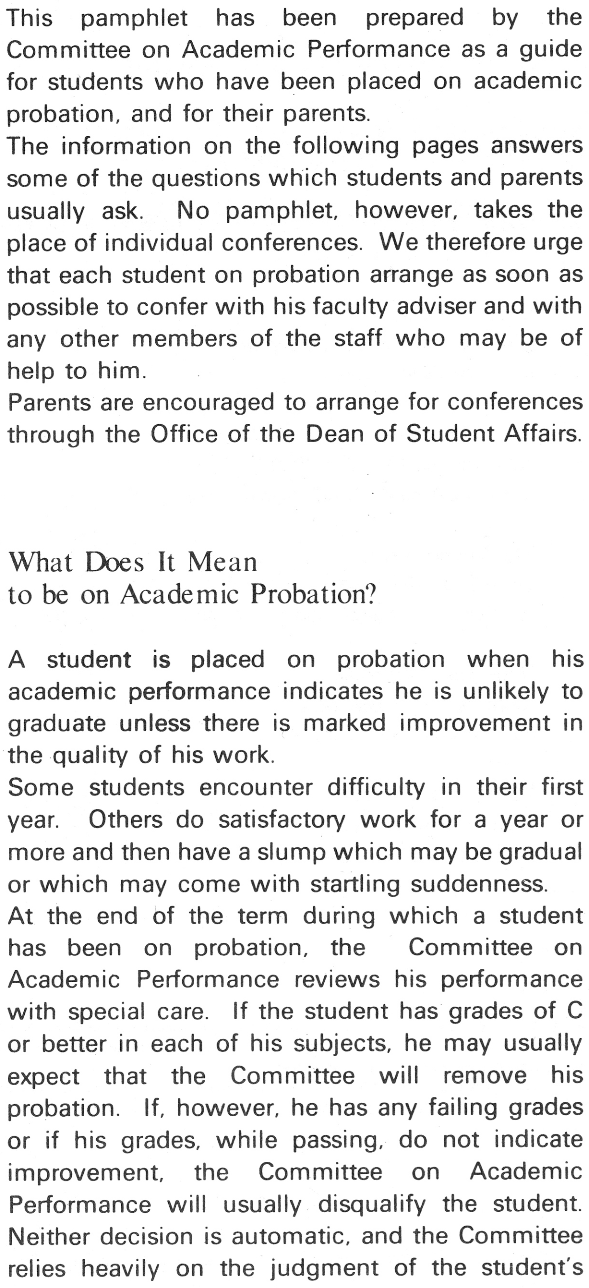
Carol Earle Chapin

This lyric relates to the decoding of Stonehenge. Gerald S. Hawkinsfed the measured positions of the stones into an **IBM 7090** computer and found that the stones marked the rising and setting of the sun and moon in midwinter and midsummer between 1500 and 2000 B.C. The final program took one minute of computer time.

This page has been automatically composed with a **PHOTON S560** photocomposing unit. The control tape for the **PHOTON** was developed by an **IBM 709** computer at the Cooperative Computing Laboratory, MIT, in about one minute.

Lyric copyrighted, reprinted by permission.

Example 3.29. *Sample of TYPRINT output.*



This pamphlet has been prepared by the Committee on Academic Performance as a guide for students who have been placed on academic probation, and for their parents.

The information on the following pages answers some of the questions which students and parents usually ask. No pamphlet, however, takes the place of individual conferences. We therefore urge that each student on probation arrange as soon as possible to confer with his faculty adviser and with any other members of the staff who may be of help to him.

Parents are encouraged to arrange for conferences through the Office of the Dean of Student Affairs.

What Does It Mean to be on Academic Probation?

A student is placed on probation when his academic performance indicates he is unlikely to graduate unless there is marked improvement in the quality of his work.

Some students encounter difficulty in their first year. Others do satisfactory work for a year or more and then have a slump which may be gradual or which may come with startling suddenness.

At the end of the term during which a student has been on probation, the Committee on Academic Performance reviews his performance with special care. If the student has grades of C or better in each of his subjects, he may usually expect that the Committee will remove his probation. If, however, he has any failing grades or if his grades, while passing, do not indicate improvement, the Committee on Academic Performance will usually disqualify the student. Neither decision is automatic, and the Committee relies heavily on the judgment of the student's

Example 3.30. *Sample of TYPRINT output. (PC6 project 8. See Section 1.9.)*

4. TYPRINT Operating Experience

4.1 Introduction

Several aspects of the operating procedures that dealt with TYPRINT input required improvement to help the user. These related to the annotation, keyboard work, editing, computer processing, and filmsetting stages. Section 4.2 discusses the technical skills needed to annotate TYPRINT source material. The convenience of the mnemonics for the control codes and ways of making interspersed coding more convenient are discussed in Sections 4.3 and 4.4, which deal with the ease of learning the codes and the speed of writing and typing them, respectively. Proofreading practice and editing programs are discussed in Sections 4.5 and 4.6. Causes of run failures, error location procedures, and error bypassing procedures are discussed in Sections 4.7 to 4.10, respectively, and some additions to the control unit which would be useful are summarized in Section 4.11.

4.2 Annotating Experience

Source material was prepared successfully for the TYPRINT option by some two dozen people of varied occupations and training. They were all able to get started without difficulty by reading the early sections of the report on which Chapter 3 is based, but they all needed a certain amount of help from the C.C.L. staff to carry the work through to completion, for reasons that are discussed in Sections 4.6 through 4.8. Most of the people were research staff working with computers but not with the typesetting project, except insofar as they used the typesetting programs to prepare material of interest to their work. About half-a-dozen nonscientists employed in editorial and other capacities in graphic arts work have annotated material successfully for TYPRINT. None of the people who have actually annotated material or keyboarded it were keyboard operators in the typesetting industry at the time, but TYPRINT conventions have been presented to senior keyboard operators in England who found no difficulty in assimilating the basic ideas. All the people who annotated manuscripts, including those with no previous experience of computers, took an interest in details of the work that they could have ignored and rapidly acquired an understanding of many of these details. It was found that editorial personnel could learn how to annotate text matter in a few hours, and that within a week of starting to use the programs they could understand the general structure of the typesetting programs and selectively suggest improvements that could be made with relative ease. Research staff had no difficulty annotating manuscripts once the typography and format had been decided, but they had considerable difficulty in specifying these. Programming staff and other personnel do not always realize the extent to which typographic design is a professional specialization. It appears to be of considerable advantage in typesetting experiments for the typography to be designed professionally—for aesthetic reasons and also because the amateur designer tends to undue complication and change of mind. The possibility of rerunning jobs with changed format is one of the advantages of computer-aided typesetting, but there are limits to the frequency that is desirable.

Accepting the use of typographic control codes that are represented by letters and digits and typed with the same keys as text characters is a conceptual jump, but it seems to present no difficulty to computer and graphic arts personnel, as stated in the preceding paragraph. The details of the control codes are of considerable importance, as they determine (1) the amount of intellectual effort needed to learn enough to start using them, (2) the intellectual difficulty of using them correctly, (3) the incremental keyboarding effort needed to type them, (4) the extent to which they are subject to errors in typing, and (5) the ease with which these errors may be detected in reading proof. Practicalities which come under these headings and which are glaringly obvious in retrospect are not always anticipated.

It is possible, and reasonably convenient, to specify typography and format for a substantial range of typesetting jobs, in terms of the functions that the control codes of the TYPRINT option represent. There are further classes of job whose typography and format cannot be specified easily, and in some cases at all, in terms of the control codes of the PC6 programs. The functions of the further codes that would be needed to deal with these are discussed in Sections 7.8 to 7.10.

4.3 Intellectual Convenience of Control Codes

The time required to annotate a manuscript depended on the time required to select the control codes and the time required to write them in. The number of control codes in the TYPRINT conventions did not seem large enough to confuse users. Several of the codes were redundant in the sense that they specified details of typography and format that could also be specified by other control codes. This was particularly true of the codes for vertical spacing. No confusion resulted from this. Different editorial personnel preferred to use different codes for the same purpose, for example to specify vertical spacing, and it seemed better to allow users to work in individual ways that they found as natural as possible, rather than to constrain all the users to adopt the working practice of just one expert.

The choice of letters for the control codes did not seem to affect their ease of use by annotators. Codes which were not obvious abbreviations of words that described their function were used as readily as codes which were. It proved much more convenient to use letters to make different codes distinct than pure numbers, as was done in the ρ-code, and was done also in the input to an interim version of the program during testing. The two letters that were used for individual control codes could have been changed with trivial effort by a programmer, essentially by altering entries in a simple table, and the program could have been adapted so that a user could change this table without any knowledge of programming. Departure from the use of pairs of capital letters as the literal part of control codes, to some other set of conventions would have required the coding of appropriate subroutines, but this would have been a relatively minor programming job that was completely independent of the design of the ρ-ω translation programs. It was not clear that users would wish to change the literal part of control codes very often with the object of facilitating annotation. One mnemonic of two letters seems as good as another, and the provision of a convention, be it arbitrary, saves the user from the effort of concocting a mnemonic. It has been noticed that editorial personnel who required special characters in material that they were setting preferred to use mnemonics that were provided in the TYPRINT manual rather than make up mnemonics themselves.

The units in which dimensions were expressed were not necessarily the most convenient for all purposes. The use of Photon points was not restrictive in any applications that were set. Provision could have been made however for other units by very simple program modifications.

4.4 Mechanical Convenience of Control Codes

As regards the time required for annotators to write control codes on a manuscript, it obviously takes longer to write six characters, as in [DL12], than to write a single special symbol that was reserved for annotating purposes. Similarly it would be quicker just to underline words that were to be italicized, than to insert disc level codes in brackets before and after every passage that is needed in italics. Reducing the number of characters in the annotations can affect the volume and difficulty of subsequent keyboard work. Where the annotations are relatively light, the total amount of annotating and keyboard work that can be saved in this way is not appreciable. For jobs in which font, indention, spacing, and so forth are changed repeatedly, measures to shorten, and if possible eliminate, the annotations may be essential for operations to be practicable. Ways in which the annotations in work such as this can be shortened, or in some cases omitted, are described later in this chapter and in Chapters 6 and 9.

The number of characters in control codes affects the number of key strokes that are needed to type them. The control codes that the TYPRINT option required were found to be rather inconvenient at the keyboard stage because of the need for numerous case shifts when they were typed on the C.C.L. Flexowriter. The left square bracket, comma, and digits were lower case, and the right square bracket and capital letters were upper case. It would have been far more convenient for both left and right delimiters to be independent of case shift, and for the case of the letters in the control codes to have been optional. The former could have been achieved in various ways, for example by putting the left square bracket on both positions of one key, and putting the right square bracket on both positions of another key, at the expense of eliminating two of the nonstandard characters that are on the Flexowriter. This illustrates the more general question of factors that affect the optimum use of keys on the keyboard perforator, which does require much further study. Some more comments on this topic are made later in Section 10.4.

4.5 Proofreading and Correcting TYPRINT Input

Keyboarding TYPRINT input on the Flexowriter produced punched paper tape and readable typescript as described in Section 3.4. The hard copy provided an opportunity for proofreading and correcting typographic errors before the computer and the typesetting equipment were used. This had some advantages, but it should be stressed that it was the experience of the C.C.L. project that proofreading the Flexowriter copy in no way removed the need to proofread the final phototypeset product. Typographic errors that were not detected when the Flexowriter copy was read quite carefully were immediately apparent on the developed Photon film. Some users adopted the attitude that jobs almost always had to be run twice, and that if only a few errors were found in the initial Flexowriter hard copy, it was simplest to correct these along with the further errors that would be found when the Photon film was inspected.

4.6 Editing Programs

Although a trained operator can produce a correct tape from a partially correct tape very efficiently by alternate tape copying and typing on a Flexowriter, the operator's attention must be directed to the job, and a Flexowriter occupied, for some time in excess of that needed for the tape to pass through the Flexowriter. For this reason, it is of considerable importance that computer typesetting programs be supplemented by adequate computer editing programs that combine the material in one tape, which contains errors, with suitably specified corrections that are provided on a second tape. The preprocessors that dealt with TYPRINT input provided rather minimal editing facilities, and this was one of the weakest features of the PC6 programs. For example, it was not permissible to have carriage-return codes within a TYPRINT editing command. Long inserts could be made despite this by rotating the platen and restoring the carriage manually, but this was a trifle irritating. It was a nuisance, too, to have to deal with each line separately, for example, when deleting several consecutive lines. The weakness of the editing program devolved largely from the lack of effort that was available for it, rather than from any fundamental difficulties in computer editing. Much more powerful editing programs have been written elsewhere (see Section 9.9), and one that was a little better in some respects was coded in the C.C.L. before the typesetting project started.[9] The basic issue that requires attention in the design of editing programs is the need to simplify the work of specifying the positions at which changes must be made. This in turn is affected by the ways in which the Flexowriter hard copy can be misleading to the proofreader. For example, line count is a reasonable way of specifying the context of a change. A proofreader determines line count by reference to visible lines of hard copy. The computer determines line count by reference to carriage return codes. A keyboard operator may press the carriage return key on the Flexowriter a few times after loading a new sheet of paper, to position it before starting to type. Each depression of the carriage-return key punches a code in the paper tape that the computer interprets as the beginning of a new line. There was particular trouble in the C.C.L. with one job, in which a keyboard operator persistently pressed the carriage-return key instead of the tabulate key, and then to "correct" the error moved the stationery back to the previous line by turning the roller and then advanced it to the appropriate lateral position by repeatedly pressing the space bar. The hard copy appeared to be correct, but the paper tape contained a carriage return followed by some twenty or thirty space codes. This particular job used a special preprocessor whose successful operation depended on the provision of the input in a definite format, in which carriage returns and space codes implied changes of font and indention. Keyboard operators must be convinced of the need to maintain a simple correspondence between what is in the tape and what is on the hard copy, and if line counts are used, these should be made from visible typed reference points. An editing program moreover should allow these reference points to be included fairly frequently to avoid lengthy counting by the proofreader who is specifying the position of an error. It should be possible also to specify the positions of changes by reference to other units of text, such as paragraphs, and in material of considerable length by hierarchical schemes that refer, for example, to chapters, sections, and pages, paragraphs, sentences, and lines. Provision should be made too for delete, insert, and replace instructions that can deal with multiline inserts and deletions, and for instructions to start new lines, paragraphs, and pages, to run on, and to transpose material. The experimental editing program that was written in the C.C.L. in 1961 did provide for such instructions, but it could deal only with texts that were too short to be of any practical interest.

The specification, design, and coding of editing programs provide scope for much further work. For example, a program that would replace every occurrence of a specified word or phrase, between specified points in a text, by some other word or phrase, would be useful and quite simple to code. Again, it would be useful to have a program that could delete a numbered equation from a mathematical text, renumber all subsequent equations, and change the references to these equations in the text, without altering any other numbers in the text (such as references to the bibliography) which coincide fortuitously with the changed equation numbers. Writing a program to do this would not be too easy.

4.7 Causes of Run Failures

TYPRINT input could contain erroneous control codes as a result of annotation errors or typographic errors in the keyboard work. Annotators quickly corrected any misconceptions that they had about the function of individual control codes or about the effects of combinations of control codes, merely by seeing the typeset output that resulted from erroneous coding. A typographic error of the annotator or keyboard operator could lead to the inclusion in the input of combinations of letters and digits which formed a valid control code that was not the one which should have been included. A typographic error also could produce a combination of letters and digits that did not form any valid code. If errors of the former sort reached the computer uncorrected, then the output was set in accordance with the erroneous control code. If an error of the latter sort reached the computer, it caused the run to terminate as soon as it was encountered during the formation of the ρ-code, under the operating procedure that was used. The treatment of erroneous control codes must be more elaborate, however, for programs to be of more practical value. There are several aspects of this problem that require attention.

In designing computer programs for almost any field of application, it is of considerable importance that provision be made to detect, identify, and record errors in the input, to limit the amount of erroneous processing that the errors can cause, and to give the user as much useful output from the run as possible, despite the occurrence of the error. The variety of errors that can occur is not always appreciated by inexperienced programmers.

4.8 Error Location Procedures

The procedures that the PC6 programs used to deal with error conditions were essentially an outgrowth of some of the provisions that were made to test the program during its development. As stated in Section 3.17, the programs could be used in either of two modes, one for recording a considerable amount of information concerning the progress of a run while the run was in progress, the other for recording only some brief comments at the end of each of a few major stages of the job. It was expedient to operate in the first mode for jobs submitted by users who were learning or experimenting with the system. The second mode was employed by users who had confidence in their familiarity with the system. When a job that was run in the second mode failed, it usually was rerun in the first mode if the cause of failure was not immediately apparent.

The TYPRINT preprocessor alternately read records of 71 Flexowriter codes from magnetic tape and translated these into ρ-code, as described in greater detail in Chapter 7. Each record of 71 Flexowriter codes was the image of the codes on a single card of the deck produced from the Flexowriter tape as described in Section 2.3. When the program was operated in the mode that produced a detailed history of the run, each record of 71 Flexowriter codes was rewritten as a string of 71 integers on the history tape, immediately before it was translated into ρ-code. If invalid data were encountered during the processing of a particular record of Flexowriter codes, the string of 71 integers which represented that record appeared as the terminal information on the history tape for the run. The error in the input then could be located by inspecting the final record of Flexowriter codes in the printout of the history tape and determining the characters that they represented by reference to a table of Flexowriter codes. It was possible very often to identify the portion of the hard copy that corresponded to this record after only a few characters had been decoded. The integer codes, however, rather than the hard copy, frequently gave the clue to the error, partly because the errors were often of a nature that did not show on the typescript. For example, the lowercase "l" and the digit "1" were indistinguishable when typed on the C.C.L. Flexowriter, and several users typed the letter "l" instead of the digit "1" in the numeric part of control codes when they prepared material for the first time. A variety of trivial measures could have been adopted to prevent users from making this particular error. It is worth mentioning here, however, as an example of the trivia that are of practical importance.

4.9 Improved Input Error Location

There are a few more general features of the way the TYPRINT preprocessor dealt with input errors, which required modification for a program to be of greater usefulness. Locating an input error by the method which has just been described presents no major difficulty, and it was done by editorial personnel who had no training in computer programming. It was unnecessarily tedious, however, and it would be much better for a typesetting program to give a precise identification of the input code at which an error condition is detected. This identification could be by the enumeration of the input record in which the error occurs and the position of the erroneous code within the record. It also could be by reference to the position in the hard copy of the character represented by the erroneous code (or the nearest character code if the erroneous code does not represent a character). The position of a character in the hard copy could be given by page, paragraph, line, word, and character counts. In the C.C.L., an error identification by record count would have been particularly useful when locating errors that were due to faulty operation of the tape-to-card converter. Translation of the erroneous code and the codes which form its context, from their integer form into the characters which they represent when these characters are in the Hollerith set (characters for which there are Hollerith codes), and into the names of characters (or of Flexowriter functions) when the codes represent characters that are not in the Hollerith set (or Flexowriter functions), would be useful.

4.10 Input Error Bypassing Procedures

A preprocessor for TYPRINT or for similarly prepared input thus should identify input errors more effectively than was done in the C.C.L. A further desirable feature of such preprocessors would be the ability to continue their operations (such as the formation of ρ-code) after erroneous input codes have been encountered. If a specimen of TYPRINT input that was processed by the PC6 programs contained several erroneous control codes that were not apparent on the hard copy, it was necessary to rerun the job several times before the errors were all found and corrected. Each of these times the run failed on the first remaining erroneous code. Even with a good turnaround time on the computer, this could lead to an excessive elapsed time for the processing of a job. In this type of work, it is far better for the portion of the program that deals with the input to record each erroneous control code as it is encountered, as described above, and then to proceed with the translation of further input material. The run could be terminated when the translation, or some initial phase such as ρ-code formation, had been accomplished or attempted for all of the input material. In this way all the invalid control codes would be detected in one computer run. Some simple convention would be needed with regard to the procedure that should be adopted when an invalid condition was encountered in the processing of a control code. The simplest such convention when dealing with TYPRINT input would be to ignore all further codes until the next left or right square bracket and to assume that this bracket was followed by text if it was a right bracket, and by a control code if it was a left bracket. This convention would skip over further control codes that occurred between the same pair of square brackets as the erroneous codes. It would also skip over text material up to the next control code or string of control codes in a pair of square brackets, in the case that the invalid condition was the omission of a right square bracket. The conventions could be made more elaborate to reduce the amount of input that might be skipped, for example by resuming the translation at the first square bracket or comma which followed the error condition. Since a comma may occur in TYPRINT input between control codes, within a section definition, or in text, the conventions in this case could become quite lengthy. This is mentioned, not because the TYPRINT control codes have a permanence that warrants any detailed discussion of error bypass procedures designed specifically for these codes, but merely to stress the fact that streamlining a system for production work requires the consideration of a very considerable amount of detail. It happens, too, that this streamlining often requires the expenditure of considerable effort to contend with the consequences of completely arbitrary choices of convention that were made at an early stage in the development of the programs. Very often the requirements for streamlined work cannot be gauged until a prototype program has been made the vehicle for trial runs, and because of lack of knowledge of the requirements for streamlining at the outset, the prototype program becomes committed to constraints that make subsequent streamlining inconvenient. For this reason, systems of programs often have to be redesigned.

Errors that were detected by the preprocessor in TYPRINT input were corrected in the ways described for errors that were found by proofreading the Flexowriter hard copy.

4.11 Control Unit Modifications

The experience of running TYPRINT and other jobs on the C.C.L. Photon suggested several features which could be added to the control unit without undue effort and which would be of considerable benefit when the operation of the Photon is interrupted and restarted, and when small reruns are made to set corrections stripped in during subsequent page make-up work. Similar features would be of use, too, on other tape-operated typesetting machines that are used in computer-aided typesetting processes.

The progress of a Photon run was monitored in the C.C.L. by a leading counter, which indicated the number of points by which the film had been advanced since the counter was last reset by the operator. The operator could stop a run by depressing a switch on the console when the counter showed a value for which he had been waiting. It would be much more convenient to have a counter that the operator could set, at the beginning of a run, to the number of points of vertical space at which the run should stop automatically. At times, it would be advantageous to be able to stop a run when the number of codes, of a type that was specified by a set of eight switches on the console, reached a value that had been preset on a counter by the operator. For the purpose of restarting a run where it had been interrupted to process another job, it would be useful to have a switch to disconnect the photographic unit from the control unit, so that the tape could be positioned automatically (and preferably at high speed) to a code whose type and position had been specified on the console by the set of switches and the counter that were also used for stopping runs. Restart procedures would be facilitated further by a set of switches on which codes could be entered manually to establish appropriate disc level, lens, and leading selection. These switches would be used to restart a tape at a point beyond that at which these conditions were specified by punched codes.

5. Preparing and Processing TABPRINT Input

5.1 Punched-Card Tables

The TYPRINT option of the PC6 programs could be used to set material in which the nature and position of typographic change were limited only by the functions which the control codes could perform. Typographic flexibility was achieved by including control codes in the input wherever they had to take effect. This could become irksome and at times completely impracticable, when repeated change was necessary. Examples 3.20 to 3.25 showed the way in which control codes tended to dominate the input when tabular material was processed with the TYPRINT option. The TABPRINT option dealt with a particular form of tabular material in a much more convenient fashion. It illustrates an approach which can be used for other forms of tabulation of numerical and verbal material.

The TABPRINT option dealt with tables that were punched in Hollerith cards in a fixed format, or written on magnetic tape in binary coded decimal records of fixed format. It is a very common practice in scientific and commercial applications of computers and conventional punched-card machines to punch the information that is to be printed on successive lines of a table into a succession of Hollerith cards, so that the same card field is used throughout the deck for items to be printed in the same column of the table. Combinations of one or more holes, whose vertical positions represent a character in the Hollerith code, can be punched in 80 columns of a Hollerith card that are referred to as columns 1 to 80, counting from left to right. A "card field" consists of some number of consecutive card columns in which an item of data is punched from left to right. A card field can consist of a single card column if it is used for an item of data that always consists of just one character. Numerical information is punched in a card field so that the same card column has the same significance relative to the decimal point throughout the deck of cards. Leading and trailing zeros in a numerical item need not be punched—an unpunched card column is regarded as a representation of zero for arithmetical purposes by the machines which process information that they read from cards. Although it is customary to punch a datum from left to right within a card field, it is quite common for the arrangement of fields across a card to differ from the arrangement which is used to print the items in these fields across a page.

Hollerith information also can be "written" on magnetic tape in a code that represents each character by a pattern of small magnetized areas on the surface of the tape, just as patterns of holes represent characters on punched cards and in punched paper tapes. The information in a punched card is represented on magnetic tape by a "record" of 80 consecutive magnetic tape codes that correspond to the 80 columns of the card. A special code on the tape corresponds to an unpunched card column. Successive records on the tape are separated by a further code that the computer can recognize. A magnetic tape record that is written with output by the computer can contain more than 80 successive codes each of which represents a character or a blank. The positions which characters occupy within a record can be numbered consecutively just as card columns are numbered 1 to 80, and a sequence of character positions within a magnetic tape record constitute a record field just as a sequence of card columns constitute a card field. The TABPRINT option of the PC6 programs could deal with tables that were written on magnetic tape in fixed format. Successive tape records had to contain data for successive lines of the typeset table. Each record could contain up to 120 characters.

Printing from cards is done conventionally on line printers which provide fixed spacing and a single style and size of type that has a rather low typographic quality. Line printers, or tabulators as they are sometimes called, can print from cards at speeds up to 150 lines per minute. Line printers can also be operated by magnetic tapes at speeds up to 1200 lines per minute. Information in punched cards can be typed mechanically by card-operated typewriters, or punched mechanically onto paper tape and then typed mechanically by typewriters that are operated by paper tape. These measures give copy whose appearance is rather better than the output of present line printers, but it is much slower and is limited as regards typographical flexibility.

To use the TABPRINT option, codes that represented the typographic style and format in which a table was to be set were punched into a few Hollerith cards in accordance with some simple conventions which are described in the next section of this chapter. The cards were called specification cards, and together they formed the specification deck. The codes for type style, point size, and leading were the same as those which were used with the TYPRINT option, and were punched in the first few specification cards. Further specification cards were punched with details of the card fields which contained the data that were to be set in the successive columns of the typeset table, and the indentions and widths of these typeset columns. The specification deck was placed in front of the card deck in which the table was punched, and the combined deck was read by the computer as input to a TABPRINT run. The computer wrote the requisite Photon codes onto a magnetic tape from which the Photon tape was punched off-line, just as in TYPRINT runs. When the TABPRINT option was used to set a table that was written on magnetic tape in binary coded decimal records of fixed format, the specification deck was prepared and read by the computer in much the same way as for a table on punched cards.

Although the TABPRINT option was intended primarily for tables that had been generated by a computer or by punched-card machines, it could be used equally for tables that had been punched on cards in fixed format by keyboard operators. The Boston University *Catalog of Lunar Craters*, which was the longest TABPRINT job that was run, in fact was prepared in this latter fashion.

Throughout the present chapter, statements concerning the preparation of tabular source material and the operation of the computer all carry the implicit qualification "within the TABPRINT conventions" and do *not* apply to computer-aided typesetting in any wider sense.

5.2 A Simple Example

The TABPRINT conventions may be introduced by reference to the way that they were used to set a table of computer-generated results for inclusion in an appendix to the Ph.D. dissertation of

R. M. Pitzer. The results were punched in a deck of table cards by the computer in one run. These cards, preceded by a specification deck of 16 specification cards, then served as input to a second computer run which used the TABPRINT option of the PC6 programs to produce the Photon operating tape. Part of the listing of the input to this second run is reproduced as Example 5.1. It includes the specification cards and the first few table cards. Successive character positions in the listing, going from left to right, correspond to successive card columns. Asterisks in the first sixteen cards were in column 1. Columns 3 to 12, 15 to 18, 20 and 21, 23 and 24, 29 to 38, 41 to 44, 46 and 47, and 49 and 50 in the table cards were used for the successive table entries, with the units digit (or sign) in columns 3, 15, 21, 24, 29, 41, 47, and 50, respectively, and columns 3, 12, 18, 20, 23, 29, 38, 44, 46, and 49 sometimes left blank.

The first page of phototypeset output that was obtained from this input is reproduced as Example 5.2. The first sixteen cards in the input deck were recognized as specification cards by the computer because they were punched with asterisks in column 1. Their purposes were as follows:

Card 1. This was the input medium card. It was punched * CARDS in Example 5.1, as the table to be typeset was punched on cards.

Card 2. This was the style card. It was punched with the codes LS6, DL3, LD2, BD8 in Example 5.1, to specify 6-point type, on 8-point body (i.e., 2 points space between lines), in the font on disc level 3.

Card 3. This was the main heading card. It provided the main heading that was centered over each page of typeset output. The letter U in Example 5.1 specified a shift to upper case beginning with the first letter of the main heading. The $ symbol reversed the case whenever it was encountered in the main heading card but it was not used in Example 5.1 so that the main heading was set as

STAGGERED HYDROGEN NUCLEAR ATTRACTION INTEGRALS

The end of the heading was marked by an asterisk.

Card 4. This was the table dimension card. It specified the following:

a. The number of sections of tabular data. The term "section" is used here, as in earlier chapters, for material which was set within a pair of margins whose positions were specified by the user. For TABPRINT input this was done by further cards in the specification deck. In Example 5.1, these were the cards 5 to 16 that are described next. Some of the portions of the typeset table

```
*CARDS
*IN,LS6,DL3,LD2,BD8*
* U,STAGGERED HYDROGEN NUCLEAR ATTRACTION INTEGRALS*
*12 SECTIONS, 30 LINES,PAGE 5.500 X 6.000*
*20,21,L,R,0.250,I*
*23,24,L,R,0.250,J*
* 3, 4,L,9, .500,*
* 5,12,L,L, .375,VALUE*
*15,16,L,R, .625,OCC.*
*17,18,L,L, .625, FACTOR*
*46,47,L,R, .625,I*
*49,50,L,R,0.250,J*
*29,30,L,R, .500,*
*31,38,L,L, .375,VALUE*
*41,42,L,R, .625,OCC.*
*43,44,L,L, .625, FACTOR*
  1.0           .5   1  1    -.0839698   2.0   5 14
   .2928733    1.0   2  2     .0184863   2.0   2 14
   .1714783     .5   4  4    -.0210038   2.0   5 13
   .2109132    1.0   5  5    -.1578178   1.0   1 15
   .1471101    2.0   1  2    -.0734964   2.0   2 15
   .0890895    1.0   2  3    -.0141323   1.0   4 16
   .0543457    2.0   4  5    -.0123433   2.0   5 16
```

Example 5.1. *TABPRINT input: the molecular integrals table. (From PC6 project 3. See Section 1.9.)*

```
                         STAGGERED HYDROGEN NUCLEAR ATTRACTION INTEGRALS

 i    j          value          occ. factor                  i    j          value          occ. factor

 1    1        1.0                  .5                       5   14        -.0839698            2.0
 2    2         .2928733           1.0                       2   14         .0184863            2.0
 4    4         .1714783            .5                       5   13        -.0210038            2.0
 5    5         .2109132           1.0                       1   15        -.1578178            1.0
 1    2         .1471101           2.0                       2   15        -.0734964            2.0
 2    3         .0890895           1.0                       4   16        -.0141323            1.0
 4    5         .0543457           2.0                       5   16        -.0123433            2.0
 5    6         .0613632           1.0                       1   16         .07957725           1.0
 1    4         .0200448           1.0                       2   16         .0504971            2.0
 2    5         .0165059           2.0                       4   15         .0350415            1.0
 1    5         .0501927           2.0                       5   15         .0399703            2.0
 2    4         .0276737           2.0                       7    7         .4801924             .5
 2    6         .0327038           2.0                       8    8         .2431342             .5
 1    7         .03743387          1.0                       7    8         .00000230           1.0
 2    7         .0344507           2.0                       7    9         .1058707            1.0
 4    8         .0174377           1.0                       8   10         .05360582           1.0
 5    8         .0177547           2.0                       7   10         .0141204            1.0
 1    8         .00248911          1.0                       8    9         .00759988           1.0
 2    8         .00246044          2.0                       7   15        -.00538038           1.0
 4    7         .00456415          1.0                       8   16         .00361166           1.0
 5    7         .00472996          2.0                       7   16         .0240810            1.0
 1    9         .3970447           1.0                       8   15         .0129435            1.0
 2    9         .2134644           2.0                       9    9         .4633817             .5
 4   10         .1155560           1.0                      10   10         .2430752             .5
 5   10         .1307018           2.0                       9   10         .1160513            1.0
 1   10         .06750669          1.0                       9   15        -.05851128           1.0
 2   10         .0457959           2.0                      10   16         .04605935           1.0
 4    9         .0438104           1.0                       9   16         .1412269            1.0
 5    9         .0534191           2.0                      10   15         .1070253            1.0
 1   11         .4427961           1.0                      11   11         .5356495             .5
```

Example 5.2. *TABPRINT output: the molecular integrals table.*

```
*CARDS
*IN,LS8,DL14,LD2,BD10*
* U,D$ISTRIBUTION OF THE $M$AJOR $O$RIGINS BY $R$EGIONS,1951*
* 6 SECTIONS,  8 LINES,PAGE 6.500 X 3.000 *
* 1,12,U,L,1.000,O$RIGIN*
*17,20,U,C,1.000,M$ARITIMES*
*22,25,U,C,1.0  ,Q$UEBEC*
*27,30,U,C,1.0  ,O$NTARIO*
*32,35,U,C,1.0  ,P$RAIRIE $P$ROVINCES*
*37,40,U,C,1.5  ,B$RITISH $C$OLUMBIA*
BRITISH         17.8  7.3 45.9 17.4 11.4
FRENCH           6.9 77.0 11.1  4.0  1.0
GERMAN           5.2  2.0 35.8 48.0  8.9
ITALIAN          2.1 22.5 57.6  6.5 11.3
NETHERLANDS     10.5  1.2 37.2 38.4 12.6
POLISH           1.3  7.7 40.9 42.5  7.4
SCANDINAVIAN     2.6  1.9 13.2 58.8 23.2
UKRAINIAN        0.3  3.3 23.7 66.9  5.7
```

Example 5.3. *TABPRINT input.*

Distribution of the Major Origins by Regions,1951

Origin	*Maritimes*	*Quebec*	*Ontario*	*Prairie Provinces*	*British Columbia*
BRITISH	*17.8*	*7.3*	*45.9*	*17.4*	*11.4*
FRENCH	*6.9*	*77.0*	*11.1*	*4.0*	*1.0*
GERMAN	*5.2*	*2.0*	*35.8*	*48.0*	*8.9*
ITALIAN	*2.1*	*22.5*	*57.6*	*6.5*	*11.3*
NETHERLANDS	*10.5*	*1.2*	*37.2*	*38.4*	*12.6*
POLISH	*1.3*	*7.7*	*40.9*	*42.5*	*7.4*
SCANDINAVIAN	*2.6*	*1.9*	*13.2*	*58.8*	*23.2*
UKRAINIAN	*0.3*	*3.3*	*23.7*	*66.9*	*5.7*

Example 5.4. *TABPRINT output corresponding to input shown in Example 5.3.*

```
                         Boston University Catalog of Lunar Craters

Seq.  Name                ξ        η          λ       β      Dia.  Q   P  Rmks.  Reg.

1451                   .0657    -.2691      3.9   -15.6     1.7  B             C6B
1452                   .1718    -.2701     10.3   -15.7     3.5  A             C5A
1453                   .1503    -.2702      9.0   -15.7     4.2  B             C5A
1454                   .1161    -.2703      6.9   -15.7     2.5  B             C5A
1455                   .1942    -.2706     11.6   -15.7     1.1  B             C5A

1456                   .1216    -.2708      7.3   -15.7     3.2  A             C5A
1457                   .2007    -.2708     12.0   -15.7     2.1  B             C5A
1458                   .2084    -.2715     12.5   -15.8     2.8  B             C5A
1459                   .2002    -.2721     12.0   -15.8     1.8  B             C5A
1460                   .0432    -.2723      2.6   -15.8     3.2  A             C5A

1461                   .1690    -.2729     10.1   -15.8     4.7  A             C6B
1462                   .0886    -.2730      5.3   -15.8     2.4  B             C6B
1463                   .1491    -.2733      8.9   -15.9     3.5  A             C5A
1464  AIRY P           .1399    -.2734      8.4   -15.9     7.4  A             C5A
1465  PARROT T         .0699    -.2736      4.2   -15.9     8.1  B             C5A

1466  PARROT S         .0598    -.2741      3.6   -15.9    10.1  C             C6B
1467                   .1950    -.2746     11.7   -15.9     2.8  A             C5A
1468                   .1508    -.2746      9.0   -15.9     2.1  C             C5A
1469                   .0962    -.2748      5.7   -16.0     2.8  B             C5A
1470                   .0818    -.2752      4.9   -16.0     7.0  B             C5A
```

Example 5.5. *Output of a later, more flexible version of TABPRINT. (From PC6 project 9. See Section 1.9.)*

```
      a(u,v,w,n,m) coefficients

u  v  w  n  m        a

1  0  5  6  1      -8/693

1  1  4  2  2       1/126
         4  2       2/385
         6  2       4/3,465

1  2  3  2  1      -1/63
         4  1      -1/770
         4  3       1/1,540
         6  1      2/693
         6  3       1/6,930

1  3  2  2  2       1/126
         4  2       1/462
         4  4      -1/9,240
         6  2      -2/3,465
         6  4      -1/41,580

1  4  1  2  1      -1/63
         4  1       3/385
         4  3       1/770
         6  1      -1/693
         6  3      -1/9,240
         6  5      -1/166,320

1  5  0  2  2       5/126
         4  2      -1/231
         4  4      -1/924
         6  2       1/2,772
         6  4       1/41,580
         6  6       1/332,640

2  0  4  0  0       1/35
         2  0       1/21
         2  2       1/126
         4  0     -16/385
         4  2       2/385
         6  0      -8/231
         6  2       4/3,465

2  1  3  2  1      -1/63
         4  1      -1/770
         4  3      -1/1,540
         6  1       2/693
         6  3      -1/6,930

2  2  2  0  0       1/105
         2  0       0
         2  2       0
         4  0      -1/55
         4  2       0
         4  4      -1/9,240
         6  0       2/231
         6  2       0
         6  4      -1/41,580
```

```
      b(n,m,u,v,w) coefficients

n  m  u  v  w                b

9  3  2  1  4        2,027,025/16
      2  1  6       -2,297,295/16
      3  0  0            3,465/16
      3  0  2         -135,135/16
      3  0  4          675,675/16
      3  0  6         -765,765/16

9  4  0  4  1          135,135/8
      0  4  3         -675,675/4
      0  4  5        2,297,295/8
      1  3  1         -135,135/2
      1  3  3          675,675/1
      1  3  5       -2,297,295/2
      2  2  1         -405,405/4
      2  2  3        2,027,025/2
      2  2  5       -6,891,885/4
      3  1  1          135,135/2
      3  1  3         -675,675/1
      3  1  5        2,297,295/2
      4  0  1          135,135/8
      4  0  3         -675,675/4
      4  0  5        2,297,295/8

9  5  0  5  0         -135,135/8
      0  5  2        2,027,025/4
      0  5  4      -11,486,475/8
      1  4  0         -675,675/8
      1  4  2       10,135,125/4
      1  4  4      -57,432,375/8
      2  3  0          675,675/4
      2  3  2      -10,135,125/2
      2  3  4       57,432,375/4
      3  2  0          675,675/4
      3  2  2      -10,135,125/2
      3  2  4       57,432,375/4
      4  1  0         -675,675/8
      4  1  2       10,135,125/4
      4  1  4      -57,432,375/8
      5  0  0         -135,135/8
      5  0  2        2,027,025/4
      5  0  4      -11,486,475/8

9  6  0  6  1        2,027,025/2
      0  6  3      -11,486,475/2
      1  5  1       -6,081,075/1
      1  5  3       34,459,425/1
      2  4  1      -30,405,375/2
      2  4  3      172,297,125/2
      3  3  1       20,270,250/1
      3  3  3     -114,864,750/1
      4  2  1       30,405,375/2
      4  2  3     -172,297,125/2
      5  1  1       -6,081,075/1
      5  1  3       34,459,425/1
      6  0  1       -2,027,025/2
      6  0  3       11,486,475/2
```

Examples 5.6 and 5.7. *The later TABPRINT output. (From PC6 project 5.*[10] *See Section 1.9.)*

that would commonly be called columns each consisted of single sections, but some of the columns consisted of pairs of sections, to obtain alignment on a decimal point.

b. The number of lines of tabular entries per page of typeset output, after which a page gap was left and the main heading and column headings repeated. The distance between page gaps was determined by the page depth which was specified separately (see item *d*, here). If the page depth exceeded the total depth of the headings and the table entries, blank film was left between the last line of table entries on a page and the standard page gap. In Example 5.1, thirty lines were specified per page.

c. The width of the area over which the main title was centered. This was the first number that followed the word PAGE. In Example 5.1 it was 5.500 inches.

d. The page depth, that is, the distance between successive page gaps in the film. In Example 5.1 it was 6 inches. The page gaps were marked in the film by holes which the Photon machine punched when it read an appropriate code in the operating tape, to guide the cutting in the darkroom. Three blank lines were left automatically between the main heading and the section headings on each page. Two blank lines were left between the section headings and the first line of table entries.

Cards 5 to 16. These were the field control cards. A separate card of this type was provided for each output section. The cards were arranged in the order in which the sections to which they related were to be set, going from left to right on the phototypeset page. Each field control card specified the following:

a. The column number of the left boundary of the card field from which the section was to be set.

b. The column number of the right boundary of this card field.

c. The case of the first character in the section heading. The letters U and L were used to specify upper and lower case, respectively.

d. The alignment of the typeset section within its margins. The letters L, C, and R were used to flush left, center and flush right, respectively.

e. The section width in inches. Successive sections were made contiguous.

f. The heading of the section (if any), with the $ symbol used as case-shift indicator.

Card 5. This set the contents of columns 20 and 21 flushed right to a margin that was $\frac{1}{4}$ inch from the left edge of the area in which setting could occur, with the heading "i."

Card 6. This set the contents of columns 23 and 24 flushed right to a margin that was $\frac{1}{4}$ inch further across the page, under the heading "j."

Card 7. This set the contents of columns 3 and 4 flushed right to a margin that was $\frac{1}{2}$ inch further across the page, without any heading.

Card 8. This set the contents of columns 5 to 12 flushed left to this last mentioned margin, under the heading "value." Cards 7 and 8 thus set the third column of output aligned on the decimal point.

Cards 9 and 10. These set the contents of columns 15 to 18, aligned on the decimal point, under the heading "occ. factor."

Cards 11 to 16. These set the last four columns of the table at the right of the output page in a similar manner to that in which cards 5 to 10 set the first four columns.

This example used an early version of the TABPRINT option. Input and output for another short table that was run with this version as a test are reproduced as Examples 5.3 and 5.4. A later version allowed the use of multiline page headings, different styles and sizes of type in different columns, and the inclusion of special symbols in page and column headings. Output that was produced with this later version is reproduced as Examples 5.5 to 5.7.

5.3 Operating Experience with TABPRINT

The TABPRINT option of the PC6 system of programs was used to set several tables that were of interest to the author and to his colleagues. Examples 5.2, 5.5 and 5.6, and 5.7, are reproduced from the publications that are numbered 3, 5, and 9, respectively, in the list at the end of Section 1.9. A separate account has been given of the production of the publication 5.[10]

The advice of personnel with experience of typographic design proved useful in the selection of style and format. The construction of the specification decks for the TABPRINT jobs that were run in the C.C.L. presented no difficulty once the decisions concerning the typography had been made. These decks were prepared in every case by programming personnel who were associated with the projects in which the table had been produced.

Although setting computer-generated numerical tables may seem one of the most obvious and straightforward applications of computer-aided typesetting, there are several factors which must be considered in this form of work, and the mechanized setting of certain numerical tables can be difficult. Some of the issues that are involved in setting tabular material by computer are discussed in the paragraphs that follow.

The setting of table headings and of table entries involve rather separate typographic considerations. It is easy to write a program to produce typesetting codes from numerical data which have been punched on cards in fixed format and which are to be set in a uniform style and size of type. For such simple tables, the programming that is needed to produce column and page headings from input typed on a simple keyboard device, such as a Flexowriter or a card punch, is much more complicated than the programming that is needed to deal with the table entries. It is expedient, therefore, when designing typesetting programs for simple tabular material, to allow the sequences of operating codes for the headings to be prepared independently and read into the computer intact at the beginning of the table setting run. The codes for the headings that have been stored can then be punched at the beginning of the codes for each page of output. The program that deals with the table then does not have to perform any of the code translation that is needed to produce the typographic niceties of column headings from input that has been punched on a simple keyboard device. The adoption of this approach with the Photon 560 would still require the use of a typesetting program to produce the sequences of Photon codes for the headings even if this program were independent of the table-setting program, since there is no keyboard perforator that punches operating tapes for this typesetting machine. It may be mentioned here, however, that another project, which is discussed in Section 8.3, uses a Linofilm phototypesetting machine for which operating tape is normally punched on a Linofilm keyboard perforator. Punched-card tables which have been produced by lengthy calculations on a computer are typeset without the need for keyboarding the computer results, by the use of a simple program which translates the table entries from the conventional bcd (binary coded decimal) form in which they are first recorded, into the code of the Linofilm typesetting machine. The heading material is typed on the Linofilm keyboard perforator, and the operating tape that results is read into the computer at the beginning of the table-setting run, so that the codes in it can be repunched in the output tape together with the operating codes for the table entries. This example of table setting illustrates the benefit of being able to merge sequences of typesetting codes which it is advantageous to generate in the computer, with sequences of codes which it is advantageous to generate on a special keyboard device. This sort of situation warrants the use in computer-aided typesetting of tape-operated equipment for which tape can also be punched directly on a special keyboard perforator. The occasional use of the special perforator does not, of course, necessitate its installation along with the phototypesetting unit if the latter is used primarily for computer-generated material. In such circumstances, the keyboard perforator in another installation could be used for the occasional needs to avoid code translation in the computer.

The production of column headings for tabular material sometimes requires two typographic control functions that the PC6 programs did not provide. It is necessary, quite often, to expand the space between two or more words that form the last line (or only line) of a column heading, to fill the width of the column. The PC6 programs expanded interword spaces only when the justification process was in operation, overset had occurred, and material was left to be set on the next line. A program extension was needed to allow a further typographic control code to be placed at the end of a portion of material that was to be followed by a line break in the typeset end product. This code would have the effect of expanding the interword spaces to fill out the measure between given margins in the typeset line.

Again, many tables require two or more column headings that each occupy more than one line. The headings for such tables could not be set by any of the variants of the PC6 programs that were written in the C.C.L. because of the dearth of page composition facilities. The further programming that is needed to provide such facilities would be quite considerable, as discussed in Section 7.10.

Computer output that is printed on line printers is often produced in quite complicated formats by programming techniques such as the use of the FORMAT statements of the FORTRAN compiler. These formats make the line printer output more readable, within the limitations of the Hollerith character set and a uniform size of type. Several useful format characteristics can be obtained in material that is phototypeset by a program with the general nature of the TABPRINT option of the PC6 system, if the input possesses these characteristics as the result of suitable programming in its production as the output of an earlier computer run. The relative expediency of obtaining the requisite typographic effects by additional programming (1) in the production of input for a program such as that which dealt with TABPRINT, and (2) in extensions to the programs that deal with this input, is a complex issue, of the sort that is discussed further in Section 10.5. Some examples of the approach (1) are described in the paragraphs that follow.

Each blank card in a TABPRINT input deck produced a blank line in the typeset output. A program which constructs and records a table in a deck of cards can be designed to leave blank cards at appropriate points in this deck. Blank lines could be produced in this way at uniform intervals (say after every five or ten nonblank lines) or whenever the values of particular tabular entries changed.

In some tables, entries in particular columns are printed in just the first line of each portion of the table in which their values stay constant, and in the first line of each new page. A program that produced TABPRINT input could be designed very easily to punch particular tabular entries only when their values changed. Processing this input under the TABPRINT option of the PC6 programs then led to an appropriate typeset end product. The entry did not need to be a number—it could be, for example, the sign that preceded a numerical entry. The TABPRINT program optionally repeated the last entry in a column of one page in the first line of the next page, if the card from which that first line was set did not contain any entry for the column in question. This situation is illustrated by the following example. Suppose that part of a deck of cards that carried a table of products of primes was punched as follows:

7	11	77
	13	91
	17	119
11	13	143
	17	187

that is, the first factor was punched in a card only when its value changed. Suppose that when this table was phototypeset using the TABPRINT option, a page break occurred between the setting of the second and third cards whose contents are given above. Without the provision for repeating entries, the output would have appeared as follows:

7	11	77	
	13	91	
———	———	———	page break
………	………	………	headings
	17	119	
11	13	143	
	17	187	

A special code could be punched in the TABPRINT specification cards for this job, however, so that entries in the first column were printed on the first line of a page, even if they were missing from the data card for that line. The output for the example under discussion would then have appeared as follows:

```
. . . . . . . . . . . . . . . . . . .
   7      11        77
          13        91
__________________________ page break
. . . . . . . . . . . . . . . . . . . headings
   7      17       119
  11      13       143
          17       187
```

In conventional computer output, each numerical result is printed as a sequence of digits that is interrupted only for a decimal point. A result is often expressed as decimal between .1 and 1 (or between 1 and 10), and a multiplicative power of 10 that is printed on the computer line printer in a convention that does not require superscripts. The possibility of setting computer results in type provides an incentive to improving the appearance of the results still further, for example by separating groups of digits by spaces or commas, deleting initial and trailing zeros, and so forth. A number of simple programming techniques of "symbol manipulation" can be used to effect these improvements in the strings of characters formed by the basic computer output subroutines. These can be incorporated in programs that produce results punched in cards which are then used as input to a table-setting program such as TABPRINT. Sections 6.3 and 6.4, describe some applications of the PC6 system that involved processes akin to those which have just been mentioned.

There are some further forms of intermittent typographic change which are needed at times in table setting and which could have been accommodated by an extension of the programs that dealt with TABPRINT input, to allow control codes to be interspersed with text characters in the table entries. A facility of this general nature would be of use, for example, to set occasional lines of a table in bold or enlarged type, as is sometimes done at uniform intervals in a table or when certain entries change value. Interspersed codes could be used also to incorporate special characters at arbitrary points in table entries. Such uses of interspersed codes could be superimposed on the use of specification cards for control information that applies throughout the bulk of a table. The total amount of coding that would be needed in the combined approach would still be far less than in a purely interspersed TYPRINT-like method.

The TABPRINT option was premised on the assumption that the decisions concerning format, and in particular section widths, would be made by the user. These decisions and the preparation of field control cards take time. It would be more convenient to have a table-setting program that determined section widths automatically from the type size, number of digits in the input field, and some simple criteria for spacing between sections. The material of the present chapter relates to tabular material which is provided in fixed format on punched cards or on magnetic tape. For the allied problem of dealing with tables that are provided in manuscript with entries of varying length, the need for mechanized format decisions concerning section widths and related matters is of even greater practical importance. This latter problem is particularly acute for technical periodicals.

6. Some Further C.C.L. Applications

6.1 BCDPRINT, Catalogs, and Built-up Formulas

The TYPRINT and the TABPRINT options of the PC6 system of programs were used in some of the typesetting experiments in the C.C.L. Typesetting applications which used other forms of input were also studied in the C.C.L. The BCDPRINT option is described in Section 6.2. It was used to set two lengthy tables of mathematical formulas which were produced mechanically on the computer by tedious algebraic processes. These two applications—the *e* coefficient tabulation and the Newcomb operator tabulation—are described in Sections 6.3 and 6.4. The production of the *International Union List of Communist Chinese Serials* also used the PC6 system. It is described in Section 6.5. The planning and testing of several further specialized applications of the PC6 system was completed in the C.C.L. One of these, the catalog, *Current Serials and Journals in the M.I.T. Libraries*, is described in Section 6.6. All the applications up to this point made use of programs that had significant differences in just the portion that is called the preprocessor. While modifying this preprocessor was sufficient for many purposes, more drastic changes were needed for some jobs, and a completely independent start for others. One job that comes in the former category dealt with *Members of the Faculty and Administration* the annual M.I.T. "Faculty Picture Book." It is discussed in Section 6.7. An application that was completely independent of the PC6 system is described in Section 6.8. This concerned some work on the phototypesetting of built-up mathematical formulas.

6.2 The BCDPRINT Option

BCDPRINT input could be typed by a keyboard operator on a Hollerith card punch[11] in much the same way that TYPRINT input was typed on a Flexowriter. The Hollerith card punch has keys for all the upper-case letters, for the digits 0 through 9, and for ten punctuation marks and special symbols, which in the C.C.L. were as follows:

. , + - * / = ( ) $

The conventions for typing BCDPRINT input can be summarized very briefly:

1. *Text characters.* These were typed using the same keys for upper- and lower-case letters. A dollar symbol was typed before the first character in a text that was to be set in upper case. Another dollar symbol was typed wherever the case had to be reversed. Dollar symbols and asterisks could not be typed as text characters. They could be included in the typeset output by use of the special character control code (CH), as described in Section 3.11.

2. *End of card spaces.* All input spaces were regarded as significant, except trailing spaces before an XS, NL, NP, or PG code. Typing was continued up to column 72 of each card, except before the codes just mentioned, and words were broken without hyphens. An input space was provided for every output space. Blank cards could be left before XS, NL, NP, and PG codes.

3. *Control codes.* The asterisk * was used both as a left and right delimiter, that is, where the left and right square brackets were used in TYPRINT input. The codes themselves were the same as in TYPRINT input.

The name BCDPRINT was given to this input and to the version of the PC6 system that dealt with it because the Hollerith codes on the cards were converted to bcd (binary coded decimal) form when they were transcribed to magnetic tape for input to the computer and when they were stored within the computer for actual processing. Moreover, material that had not been keyboarded could be generated by the computer and written on magnetic tape in bcd form for processing by the BCDPRINT programs.

An illustration of BCDPRINT input is given as Example 6.1. It was rather tedious to prepare manually. The main use of the BCDPRINT option was for Hollerith material that was generated by the computer.

```
*IN,LS10,DL2,JU,*$CHEMISTRY *NL,VL2*T$HE $D$EPARTMENT OF $C$HEMISTRY OFF
ERS A SINGLE UNDERGRADUATE $C$OURSE, SUFFICIENTLY FLEXIBLE IN ITS ELECTI
VES SO THAT IT PROVIDES EXCELLENT PREPARATION FOR CAREERS IN MANY DIFFER
ENT AREAS OF CHEMISTRY, AND GRADUATE PROGRAMS FOR THREE ADVANCED DEGREES
.  $T$HERE ARE EXCELLENT OPPORTUNITIES FOR STUDY AND RESEARCH IN PHYSICA
L, ORGANIC, NUCLEAR, AND ANALYTICAL CHEMISTRY. $I$N ADDITION, THE $D$EPA
RTMENT IS RESPONSIBLE FOR UNDERGRADUATE AND GRADUATE INSTRUCTION IN CHEM
ISTRY FOR STUDENTS IN MANY OTHER $I$NSTITUTE $C$OURSES. *VL5,EN*
```

The Department of Chemistry offers a single undergraduate Course, sufficiently flexible in its electives so that it provides excellent preparation for careers in many different areas of chemistry, and graduate programs for three advanced degrees. There are excellent opportunities for study and research in physical, organic, nuclear, and analytical chemistry. In addition, the Department is responsible for undergraduate and graduate instruction in chemistry for students in many other Institute Courses.

Example 6.1. *Manual input and the resulting output for BCDPRINT.*

It should be noted that the BCDPRINT cards were punched in Hollerith code, which is quite different from the Flexowriter code preserved in the computer input cards which were prepared from TYPRINT Flexowriter tapes (see Section 2.3). The preservation of the Flexowriter codes involved a *non*standard operation of the IBM 047. It would also have been possible to punch cards coded in a manner very similar to BCDPRINT from TYPRINT Flexowriter tapes, by operating the IBM 047 in the manner for which the equipment was designed. It would have been far better to have done this when processing TYPRINT material instead of punching binary cards, and to have reformulated the TYPRINT and BCDPRINT conventions slightly and used a single program to deal with Hollerith input that was either (1) keyboarded on a card punch, (2) punched mechanically from Flexowriter tape on an 047 or similar machine with Flexowriter code to Hollerith code conversion, or (3) generated mechanically in the computer and punched on cards, or written on magnetic tape.

6.3 The *e* Coefficient Table

The typesetting applications which are considered in this section and in Sections 6.4 and 6.8 relate to the use of computers to perform tedious algebraic processes mechanically.[12,13,14] Most conventional applications of computers form numerical results from numerical data by arithmetical processes. In contrast, however, applications such as that considered here produce formulas from other formulas by algebraic operations. These two types of process are illustrated respectively by the addition of 1 and 2 to form 3 (arithmetic) and the addition of $x+y$ and x-$2y$ to form $2x$-y (algebra), respectively. Computers now are being used to a rapidly increasing extent for algebraic work, such as the production of the *e* coefficients which are described here, and for numerous other types of "nonnumeric" process.

The *e* coefficients are algebraic expressions which are used in calculations that arise in theoretical research in molecular quantum chemistry. About 1000 of these *e* coefficients are needed for some calculations that are of current interest. Each *e* coefficient is labeled by four indices, and the coefficients can be listed in a standard order for reference. The first three of the *e* coefficients in this list are

2
2k/(2k+1)
2(k+1)/(2k+1)

but they become more complicated later. Thus the 102nd is

$$2(k+1)(2k^4+8k^3-7k^2-30k+18)/\ \{(2k-1)(2k+1)(2k-3)(2k+5)(2k+7)\}$$

In mathematical terminology, the e coefficients are all rational algebraic expressions in a variable which is denoted by k. A computer program was used in the C.C.L. to generate expressions for the e coefficients in an internal computer code. A further program then produced the BCDPRINT representation of these expressions which was then converted by the PC6 system of programs into the Photon operating codes. A portion of the phototypeset output is reproduced as Example 6.2 with the BCDPRINT representation of the last three lines of this portion as Example 6.3. Some of the e coefficients are too long to be set on a single line. The program which produced the BCDPRINT representation tested the internal computer form of each coefficient to determine its length. If the coefficient was too long to go on a single line, the program formed the BCDPRINT representation that put the numerator on one line and the denominator on another. No numerator or denominator ever exceeded one line by itself. The maximum length of the expressions had been found by inspecting a crude printout of the formulas on the computer line printer before the program that formed the BCDPRINT representation was written. A sample of this crude computer printout is reproduced as Example 6.4. It does not have the typographic quality of the phototypeset table which was produced subsequently. The program which produced the BCDPRINT input from the internal computer representation took a member of the C.C.L. research staff about a month to write, working on it part time. The prior work had gone on for several months. The production of the internal computer representation of the table of coefficients took 2 hours of 709 time. The production of the BCDPRINT input from this took $\frac{1}{2}$ hour of 709 time. The conversion of this to the Photon codes by the PC6 system took another $1\frac{1}{2}$ hours of 709 computer time. Setting the 21-page table of formulas took 6 hours of production time on the Photon machine. The first 2 pages of the table had taken 10 weeks to produce by hand some years before. This part of the table took the computer 5 minutes to construct. The e coefficients get progressively more difficult to construct as the table continues, and an error at any point makes all the subsequent coefficients wrong since each coefficient is constructed from its predecessors in the list.

The preliminary computer printout (Example 6.4) is readable despite the limitations of the line printer since it requires just the letter K, the symbols +, -, and /, and parentheses (and), and the exponents can be distinguished even though they are the same size as the main characters. The phototypeset version, however, is much easier to read and is far less bulky. The personnel and machine times that were needed to write and test the BCDPRINT input generator for this job and to perform the computer and the Photon runs which pertained to the typesetting were entirely appropriate to the tempo at which research work is conducted. Converting the initial computer results to Photon code mechanically was preferred to keyboard setting from the crude computer printout, as it precluded transcription errors and the need for detailed proofreading. In general, when a research project, be it large or small, is apparently completed in substance, weeks or months of further effort can go into "rounding-off" operations that may have a substantial relative effect on the significance and usefulness of the project. From the standpoint of printing economics, it might seem an extravagance to a printer for a skilled research worker to spend a few weeks writing and testing programs to set a 21-page table of formulas. This may be compared however with an author spending a few more weeks revising a 21-page manuscript. Viewed in perspective as just one more rounding-off operation in a research laboratory, the few weeks are not at all untoward. It so happened that the Photon tape was used to control a typesetting machine in the computational laboratory which provided the environment for the research work which produced the tape. Tapes for similar jobs could well be produced by research workers in thousands of other laboratories, for job setting by typesetting firms with tape-operated equipment. The sort of activity in which a six-hour typesetting run is required at the end of a several month research project cannot have an economic impact on the typesetting industry unless there are many research projects in progress. The normal text typesetting at the end of a several month research project, when results are published as a conventional paper, does not require more than a few hours of keyboard setting. But setting scientific periodicals is a substantial source of revenue to some firms, because there are so many research projects. Setting computer-generated tables and formulas is a relatively small research requirement at present. It can grow to being a requirement of a significant section of the research community that uses computers.

The details that required attention in the production of the BCDPRINT input from the internal computer representation of the *e* coefficients may be mentioned briefly. To build up the BCDPRINT string, codes were provided for (1) a left brace to start the numerator; (2) the numerator factor that did not require brackets, such as 2, -3,

u+v	w	j	i	$e^{k,j,i}_{u+v,w}$
6	0	4	6	$\{-2(k+5)(k+6)\} \div \{(2k+1)(2k+3)(2k+5)(2k+7)(2k+9)(2k+11)\}$
6	0	6	-6	$\{2\} \div \{(2k+1)(2k-1)(2k-3)(2k-5)(2k-7)(2k-9)\}$
6	0	6	-4	$\{-12\} \div \{(2k-1)(2k+1)(2k-3)(2k+3)(2k-5)(2k-9)\}$
6	0	6	-2	$\{30\} \div \{(2k+1)(2k-1)(2k+3)(2k-5)(2k+5)(2k-7)\}$
6	0	6	0	$\{-40\} \div \{(2k-1)(2k-3)(2k+3)(2k-5)(2k+5)(2k+7)\}$
6	0	6	2	$\{30\} \div \{(2k-1)(2k+1)(2k-3)(2k+3)(2k+7)(2k+9)\}$
6	0	6	4	$\{-12\} \div \{(2k+1)(2k-1)(2k+3)(2k+5)(2k+7)(2k+11)\}$
6	0	6	6	$\{2\} \div \{(2k+1)(2k+3)(2k+5)(2k+7)(2k+9)(2k+11)\}$
0	7	0	-7	$\{2k(k-1)(k-2)(k-3)(k-4)(k-5)(k-6)\} \div \{(2k-1)(2k+1)(2k-3)(2k-5)(2k-7)(2k-9)(2k-11)\}$
0	7	0	-5	$\{14k(k-1)(k-2)(k-3)(k-4)(k^2-4k-8)\} \div \{(2k+1)(2k-1)(2k-3)(2k+3)(2k-5)(2k-7)(2k-11)\}$
0	7	0	-3	$\{42k(k-1)(k-2)(k^4-4k^3-12k^2+32k+54)\} \div \{(2k-1)(2k+1)(2k-3)(2k+3)(2k+5)(2k-7)(2k-9)\}$
0	7	0	-1	$\{70k(k^6-20k^4+106k^2-132)\} \div \{(2k-3)(2k+3)(2k-5)(2k+5)(2k-7)(2k+7)(2k+1)\}$
0	7	0	1	$\{70(k+1)(k^6+6k^5-5k^4-60k^3+k^2+138k-45)\} \div \{(2k+1)(2k-1)(2k-3)(2k-5)(2k+5)(2k+7)(2k+9)\}$
0	7	0	3	$\{42(k+1)(k+2)(k+3)(k^4+8k^3+6k^2-40k+15)\}$ $\div \{(2k-1)(2k+1)(2k-3)(2k+3)(2k+5)(2k+9)(2k+11)\}$

Example 6.2. *Computer-generated* e *coefficients printed by BCDPRINT.*

```
                                                         *VP16**XS21**
FR* 0*XS2**FR* 7*XS3**FR* 0*XS4**FR* 1*XS7**FL**DL
12,34,DL3*70(K+1)(K*LS36*6*LS10*+6K*LS36*5*LS10*-5
K*LS36*4*LS10*-60K*LS36*3*LS10*+K*LS36*2*LS10*+138
K-45)*DL12,79,DL12**DL12,78,DL12**DL12,34,DL3*(2K+
1)(2K-1)(2K-3)(2K-5)(2K+5)(2K+7)(2K+9)*DL12,79,DL3
**VP16**XS21**FR* 0*XS2**FR* 7*XS3**FR* 0*XS4**FR*
 3*XS7**FL**DL12,34,DL3*42(K+1)(K+2)(K+3)(K*LS36*4
*LS10*+8K*LS36*3*LS10*+6K*LS36*2*LS10*-40K+15)*DL1
2,79,DL3**VP12**XS27**FL**DL12;78,DL12**DL12,34,DL
3*(2K-1)(2K+1)(2K-3)(2K+3)(2K+5)(2K+9)(2K+11)*DL12
,79,DL3**VP16*
```

Example 6.3. *Computer-generated output which was, in turn, the input to BCDPRINT for the last three lines of Example 6.2.*

```
                            4     3       2
6    2    4    4       -4(2K  +44K  +331K  +949K+705)/(2K-1)(2K+1)(2K-3)(2K+3)(2K+5)(2K+7)(2K+11)(2K+13)

                                     2
6    2    4    6       2(K+5)(K+6)(4K  +44K+103)/(2K+1)(2K-1)(2K+3)(2K+5)(2K+7)(2K+9)(2K+11)(2K+15)

6    2    4    8       -2(K+5)(K+6)(K+7)(K+8)/(2K+1)(2K+3)(2K+5)(2K+7)(2K+9)(2K+11)(2K+13)(2K+15)

6    2    6   -8       2(K-6)(K-7)/(2K+1)(2K-1)(2K-3)(2K-5)(2K-7)(2K-9)(2K-11)(2K-13)

                            2
6    2    6   -6       -2(4K  -56K+193)/(2K-1)(2K+1)(2K-3)(2K+3)(2K-5)(2K-7)(2K-9)(2K-13)
```

Example 6.4. *Crude computer printout of* e *coefficients.*

```
3    3    1    -4      {2(k-1)(k-2)(k-3)^2}÷{(2k-1)(2k+1)(2k-3)(2k+3)(2k-5)(2k-9)}
4    3    0    -1      {2k(3k^6-64k^4+401k^2-700)}÷{(2k-3)(2k+3)(2k-5)(2k+5)(2k-7)(2k+7)(2k+1)}
```

Examples 6.5 and 6.6. e *coefficients, showing the suppression of unnecessary expressions by the computer.*

k, 4k, -6k, if the coefficient contained such a factor other than unity; (3) the linear factors of the form $ak+b$ with nonzero a and b, if any, in brackets; (4) the nonlinear factor if present; (5) a right brace to end the numerator; (6) a division symbol; (7) to (11) codes for the denominator corresponding to (1) to (5). The codes (2) were provided by a portion of the program which took slightly different paths when the internal representation of the unbracketed factor was 1, -1, k, -k, or of the form n, $-n$, nk, $-n$k where n stands for an integer. These were stored internally, in effect as $+1k^0$, $-1k^0$, $+1k^1$, $-1k^1$, $+nk^0$, $-nk^0$, $+nk^1$, $-nk^1$, that is, as the integer pairs (1,0), (-1,0) (1,1), (-1,1), (n,0), ($-n$,0), (n,1), ($-n$,1). The different paths suppressed the leading + symbol, the factor 1, and the factor k^0, and inserted the codes for the letter k when necessary. The codes (3) were provided by a portion of the program which suppressed the leading + symbol, and inserted the code for the letter k after its numerical multiplier and an exponent outside the bracket when necessary, as in $e(3,3,1,-4)$ (see Example 6.5). The codes (4) were provided by a portion of the program which suppressed the leading + symbol and terms with zero coefficients, as in $e(4,3,0,-1)$ (see Example 6.6), and which put in codes for exponents. Commas were inserted between groups of 3 digits in coefficients that exceeded 999. The denominator was treated in much the same way as the numerator. A test was made on the over-all length of the expression for the e coefficient, and codes for a new line were put before the division symbol if it was found that it was too long for a single line. A refinement which was not made was to arrange the factors of the form $(2k+b)$ in order of increasing independent term b.

The e coefficients are defined by

$$e(x, w, k, j, i) = (2k + 2i + 1)\frac{(k - j)!}{(k + j)!}$$

$$\int_0^{\pi} \sin^x \theta \cos^w \theta P_{k+i}(\cos \theta) P_k^j(\cos \theta) \sin \theta \, d\theta$$

The notations $P_n(\cos \Theta)$ and $P_n^m(\cos \Theta)$ are used here for ordinary and associated Legendre functions. The e(x, w, k, j, i) are zero unless x,w,j,i obey certain restrictions. The nonzero e's for increasing integer values of x,w,j,i are found by a recurrence procedure from the single starting value

$$e(0, 0, k, 0, 0) = 2$$

Each e coefficient was first constructed in an internal representation which corresponded to its formulation as the ratio of two polynomials. The representation consisted of a string of integers. The first integer was the order of the numerator polynomial. The coefficients of the successive powers of k in the numerator polynomial followed, and then the order of the denominator polynomial and the coefficients of the successive powers of k which it contained. These representations were manipulated by subroutines which formed the representations of rational algebraic expressions that were the sum, difference, and product of other rational algebraic expressions which were represented in the same manner. The subroutines were used by a program which generated the representations of the *e* coefficients by a recurrence procedure. For each *e* coefficient in turn, the linear factors of the numerator were extracted by applying the remainder theorem to every factor of the form $a\mathrm{k}+b$, where a was a factor of the high-order coefficient and b was a factor of the independent term. When a factor was found, it was extracted and the process repeated on the result. The linear factors of the denominator were extracted in the same way. The number of factors, and the linear coefficient and independent term in each, were stored as an array of integers. These processes and the formation of the BCDPRINT input were programmed in FORTRAN, using some special subroutines that perform basic manipulations of binary coded decimal characters in the latter phase of the work.[15]

It may be mentioned incidentally that the *e* coefficients proved the downfall of the first typesetting program which was written in the C.C.L. in 1961. The internal representations had been generated at that time, but the typesetting program did not allow justification to be suppressed after overset. Spaces were left between right and left brackets, and the result was differing spaces between bracketed expressions in different entries.

6.4 The Newcomb Operator Table

A table of lengthy mathematical expressions for the Newcomb operators of planetary theory was constructed and typeset mechanically in the C.C.L.[16, 17] The techniques that were used were very similar to those which were employed for the chemical research project described in the preceding section. The scientific background to the Newcomb operator work concerns the mathematical theory which is used to predict the movement of the planets. This theory must take account of the way in which the planets perturb each other's paths. The orbits are determined primarily by the sun's gravitational field, but the perturbing effects of other planets are significant, particularly in the case of the relatively small minor planets (often called asteroids) which are strongly affected by the planet Jupiter. The prediction of the long-term behavior of the orbits of certain minor planets is beset with mathematical difficulties, and the Newcomb operators which were produced in the C.C.L. and phototypeset mechanically are being used for further investigations of this and related problems.

The Newcomb operators are mathematical expressions that are labeled by two indices and can be arranged in a standard order for reference. Three forms of Newcomb operator, which are called J, X, and Ξ operators for reference, were produced. The first three operators in the table of J's are

$$J_{0,0}^{n,k} = 1$$

$$J_{1,0}^{n,k} = 2k - n$$

$$J_{2,0}^{n,k} = 5k + 4k^2 + (-3 - 4k)n + n^2$$

but the expressions become much longer as the table progresses; for example the 21st entry goes on for 10 lines. They are all polynomials in two variables, n and k. The J's have integer coefficients. The X's and Ξ's are polynomials in the same variables n and k, and have non-integer coefficients.

Simon Newcomb, the American astronomer who developed the use of the expressions which now bear his name, constructed the first 25 of the J's which were reconstructed mechanically in the C.C.L., and another 25 related expressions, at the end of the nineteenth century. The work was repeated by hand some 60 years later, by a Soviet astronomer, Sharaf, who found a few errors in Newcomb's results. These pencil-and-paper derivations of the expressions took a considerable amount of time. The effort needed to construct the expressions for a Newcomb operator goes up roughly as the fourth power of its position in the table. For investigations of current interest, expressions are needed for hundreds of these operators, particularly the X's and Ξ's.

Computer programs were written in the C.C.L. to generate expressions for the J's, X's, and Ξ's in an internal computer representation. Further programs were written to convert these to the BCDPRINT representation of the expressions that were to be set in type. The BCDPRINT representation then was converted into Photon codes by the PC6 programs. Portions of the phototypeset output are reproduced as Examples 6.7 to 6.9, together with part of the BCDPRINT material which produced one of these, as Example 6.10.

The production of the internal computer representation of some 50 pages of tables took about $\frac{3}{4}$ hour of 709 time. The reconstruction of the results which took Newcomb and Sharaf months of work required less than 10 minutes of computer time. The conversion of the internal computer representation into BCDPRINT input took $1\frac{1}{2}$ hours of 709 time, and the conversion of this to Photon codes by the PC6 programs took another $1\frac{1}{2}$ hours. Setting the film took 7 hours of Photon production time.

The J's, X's, and Ξ's are all typeset as polynomials in n, in which the coefficients are polynomials in k. The formation of the BCDPRINT representation required attention to various details which parallel those of the *e* coefficient job described in Section 6.3. Leading + symbols, unit coefficients, zero terms, and unit exponents were suppressed. Each term of the n polynomial was started on a new line if it would have overset the preceding line. The numerical coefficients in the X's and Ξ's were printed as floating-point numbers. The BCDPRINT generator used the FORTRAN I.O. subroutines to convert the internal binary form of these numbers into binary coded decimal in E format. A further subroutine converted the E format exponent into codes to set a sequence of characters of the form $\times 10^{\mathrm{p}}$, with the appropriate exponent standing in place of p when this was neither 0 or 1. The characters ×10 were included when the exponent was 1. Commas were interposed between groups of three digits in the coefficients in the J's.

It can be seen that in jobs such as this, there may be considerable scope for programming to convert the initial computer results into a representation of the characters which are actually to be set. Even with the Newcomb operator tabulation some further work could have been done. Thus, the terms of the n polynomial were all joined by + symbols, so that many of the bracketed k polynomials consist entirely of negative terms. It would have been possible to precede a term of the n polynomial by a - symbol, if the high-order coefficient of the k polynomial as originally produced were negative, and to reverse the sign of the terms in the k polynomial. This would have set $J_{2,\,0}^{n,\,k}$ as

$$5k+4k^2-(3+4k)n+n^2$$

instead of as

$$5k+4k^2+(-3-4k)n+n^2$$

Again, the library subroutine that formed the binary coded decimal representation of floating-point numbers could be improved. Round-off facility was not perfect, and as a result the X's and Ξ's as set at present contain coefficients 0.09999999 ×10 instead of 1. Trailing

$$
\begin{aligned}
X_{7,3}^{n,k} = {} & -0.33461475\text{x}10^{-1}k-0.14399801k^{2}-0.24342602k^{3}-0.22804879k^{4}-0.16070420k^{5} \\
& -0.87936740\text{x}10^{-1}k^{6}-0.31835937\text{x}10^{-1}k^{7}-0.67336309\text{x}10^{-2}k^{8} \\
& -0.74404761\text{x}10^{-3}k^{9}-0.33068783\text{x}10^{-4}k^{10} \\
& +(0.35710151\text{x}10^{-1}+0.12827432k+0.24294291k^{2}+0.24131731k^{3}+0.15108936k^{4} \\
& +0.82103588\text{x}10^{-1}k^{5}+0.35698784\text{x}10^{-1}k^{6}+0.94204695\text{x}10^{-2}k^{7} \\
& +0.12648809\text{x}10^{-2}k^{8}+0.66137566\text{x}10^{-4}k^{9})n \\
& +(-0.10648664\text{x}10^{-1}-0.11802842\text{x}10^{-1}k-0.17678235\text{x}10^{-1}k^{2} \\
& +0.90720847\text{x}10^{-2}k^{3}+0.21425374\text{x}10^{-1}k^{4}+0.77473957\text{x}10^{-2}k^{5} \\
& +0.29658564\text{x}10^{-3}k^{6}-0.22321428\text{x}10^{-3}k^{7}-0.24801587\text{x}10^{-4}k^{8})n^{2} \\
& +(-0.52983682\text{x}10^{-2}-0.16105078\text{x}10^{-1}k-0.13099048\text{x}10^{-1}k^{2} \\
& -0.19973415\text{x}10^{-1}k^{3}-0.15661169\text{x}10^{-1}k^{4}-0.50636573\text{x}10^{-2}k^{5} \\
& -0.69444443\text{x}10^{-3}k^{6}-0.33068783\text{x}10^{-4}k^{7})n^{3} \\
& +(0.29511305\text{x}10^{-2}-0.20363136\text{x}10^{-2}k-0.73852539\text{x}10^{-2}k^{2}-0.15620478\text{x}10^{-2}k^{3} \\
& +0.96752025\text{x}10^{-3}k^{4}+0.34722222\text{x}10^{-3}k^{5}+0.28935185\text{x}10^{-4}k^{6})n^{4} \\
& +(-0.26016800\text{x}10^{-2}+0.10357892\text{x}10^{-2}k+0.30802408\text{x}10^{-2}k^{2} \\
& +0.11049623\text{x}10^{-2}k^{3}+0.10850694\text{x}10^{-3}k^{4})n^{5} \\
& +(0.13163248\text{x}10^{-2}+0.29613353\text{x}10^{-3}k-0.29342086\text{x}10^{-3}k^{2}-0.97656249\text{x}10^{-4}k^{3} \\
& -0.72337963\text{x}10^{-5}k^{4})n^{6} \\
& +(-0.29522931\text{x}10^{-3}-0.10178985\text{x}10^{-3}k+0.31001984\text{x}10^{-5}k^{2} \\
& +0.20667989\text{x}10^{-5}k^{3})n^{7} \\
& +(0.31970796\text{x}10^{-4}+0.91068327\text{x}10^{-5}k+0.38752480\text{x}10^{-6}k^{2})n^{8} \\
& +(-0.16469803\text{x}10^{-5}-0.25834987\text{x}10^{-6}k)n^{9}+0.32293733\text{x}10^{-7}n^{10}
\end{aligned}
$$

Example 6.7. *Newcomb operators generated by computer and printed by BCDPRINT.*

$$
\begin{aligned}
J^{n,k}_{5,3} = {} & 688k-6{,}164k^2-9{,}586k^3-1{,}352k^4-4{,}380k^5-6{,}032k^6-2{,}240k^7-256k^8 \\
& +(1{,}296+2{,}032k+8{,}768k^2+3{,}919k^3-880k^4+1{,}856k^5+1{,}536k^6+256k^7)n \\
& +(-1{,}084+642k-1{,}671k^2+2{,}695k^3+4{,}480k^4+1{,}488k^5+128k^6)n^2 \\
& +(-168+275k+2{,}610k^2-180k^3-960k^4-192k^5)n^3+(1{,}009-385k-1{,}535k^2-420k^3)n^4 \\
& +(-720-79k+240k^2+48k^3)n^5+(202+47k-8k^2)n^6+(-24-4k)n^7+n^8
\end{aligned}
$$

$$
\begin{aligned}
J^{n,k}_{4,4} = {} & -7{,}140k^2+7{,}801k^4-2{,}272k^6+256k^8+(720+2{,}012k^2-2{,}720k^4+512k^6)n \\
& +(-1{,}044-3{,}674k^2+2{,}128k^4-256k^6)n^2+(-140+2{,}268k^2-576k^4)n^3 \\
& +(889-958k^2+96k^4)n^4+(-560+192k^2)n^5+(154-16k^2)n^6-20n^7+n^8
\end{aligned}
$$

Example 6.8

$$
\begin{aligned}
\Xi^{n,k}_{6,1} = {} & 0.42491319k+0.13270400k^2-0.58263887k^3-0.52940538k^4-0.17751735k^5 \\
& -0.26041666\text{x}10^{-1}k^6-0.13888888\text{x}10^{-2}k^7 \\
& +(-0.22146267-0.15382270\text{x}10k-0.13371744\text{x}10k^2-0.21294488k^3+0.11219618k^4 \\
& +0.40104166\text{x}10^{-1}k^5+0.34722222\text{x}10^{-2}k^6)n \\
& +(0.10800021\text{x}10+0.23384766\text{x}10k+0.13933919\text{x}10k^2+0.27007379k^3 \\
& +0.77610213\text{x}10^{-10}k^4-0.31249999\text{x}10^{-2}k^5)n^2 \\
& +(-0.78080512-0.97878689k-0.35248481k^2-0.35156250\text{x}10^{-1}k^3 \\
& +0.86805555\text{x}10^{-3}k^4)n^3 \\
& +(0.20079210+0.14930555k+0.27018229\text{x}10^{-1}k^2+0.43402778\text{x}10^{-3}k^3)n^4 \\
& +(-0.21755642\text{x}10^{-1}-0.83007813\text{x}10^{-2}k-0.39062499\text{x}10^{-3}k^2)n^5 \\
& +(0.94401041\text{x}10^{-3}+0.10850694\text{x}10^{-3}k)n^6-0.10850694\text{x}10^{-4}n^7
\end{aligned}
$$

Examples 6.8 and 6.9. *Newcomb operators printed by BCDPRINT.*

```
*XS21,LS48*    N,K*XS21,LS10,DL15,27,DL3,LS48* 6, 1*LS10,XS3*=*XS2*0.424
91319K+0.13270400K*LS36*2*LS10*-0.58263887K*LS36*3*LS10*-0.52940538K*LS3
6*4*LS10*-0.17751735K*LS36*5*LS10**VP2,XS22*-0.26041666X10*LS36*-1*LS10*
K*LS36*6*LS10*-0.13888888X10*LS36*-2*LS10*K*LS36*7*LS10**VP2,XS22*+(-0.2
```

Example 6.10. *A portion of the BCDPRINT input representation of Example 6.9.*

zeros could have been suppressed, to give, e.g., 0.625 instead of 0.62500000, but the circumstances of the project did not permit the expenditure of the further few weeks of elapsed time that would have been necessary to do all this.

The Newcomb operator tables illustrate the fallibility of present equipment and personnel. Some entries in the table were set erroneously when first run on the Photon. The errors were detected on inspection of the film. The erroneous entries were rerun, but were not stripped in, and this was not noted when the proofs were inspected, alas on New Year's Day.

As regards the formation of the Newcomb operators, they were stored in the computer as arrays of numerical coefficients. These arrays were manipulated by subroutines which performed the basic processes of addition, subtraction, multiplication, and substitution on polynomials which were represented in this array form. The mathematical procedure for generating Newcomb operators of the J variety consisted of the repeated application of a rather pernicious recurrence formula which required summation over operators of lower order. The X and Ξ operators were derived from the J's by linear transformations.

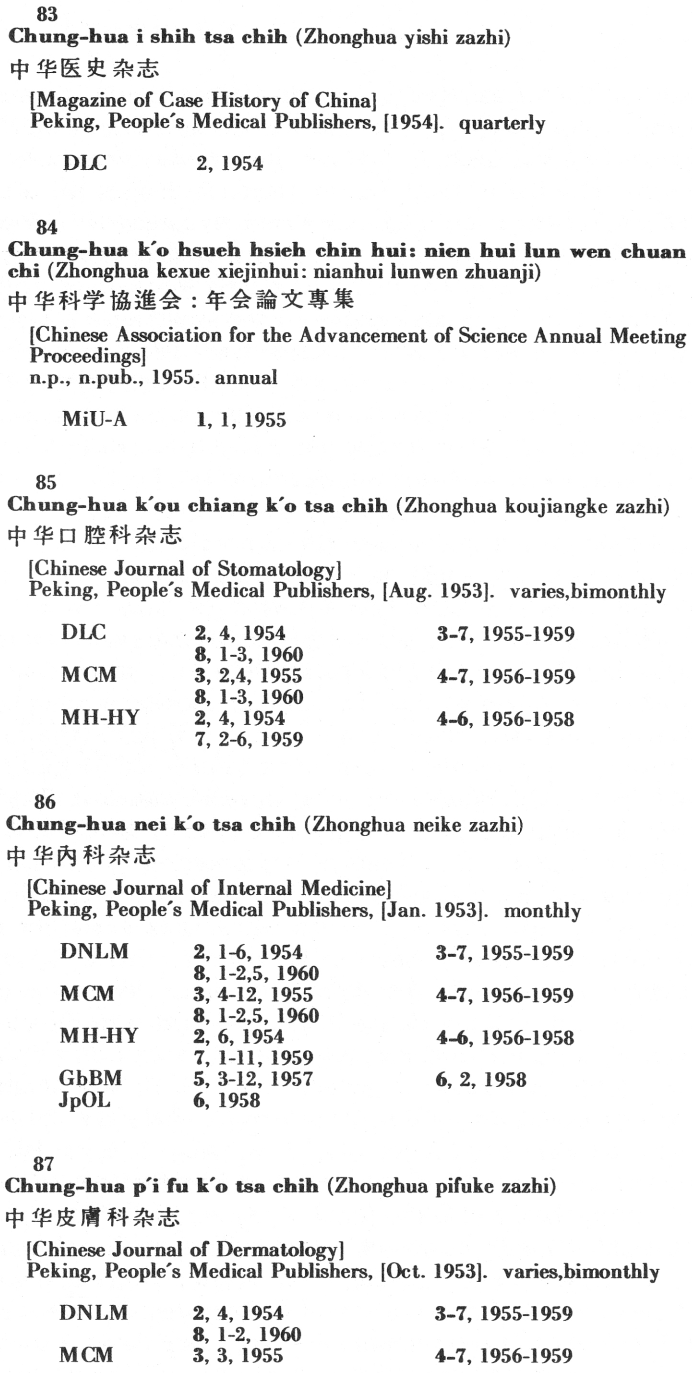
6.5 The Union List

The International Union List of Communist Chinese Serials was produced by the collaborative efforts of members of the staff of the M.I.T. Libraries and Cooperative Computing Laboratory, who were responsible for the design and bibliographic work and for the computer work, respectively.

A union list is a catalog of publications of some variety, which lists the libraries that hold each of these publications. Union lists are of obvious importance to users of published works which are held in only a few libraries, because they are rare or difficult to obtain or are highly specialized. The union list which was set in the C.C.L., and which we call the *Union List* for brevity, records the holdings of over 500 recent Communist Chinese scientific, technical, and medical serials in 28 libraries of the United States, Canada, Great Britain, Japan, and Hong Kong. It also contains a select list of 100 social science titles. For the purpose of this list, a serial is defined as a publication which is intended to continue indefinitely and which is issued under a common title, in parts whose order can be determined from numbers or dates appearing in each issue. Newspapers are excluded, and only titles which have been published on the Chinese Mainland since 1949 are included.

Two portions of the *Union List* that relate mostly to medical journals are reproduced to illustrate the design of the publication (Examples 6.11 and 6.12). Serials are listed in the alphabetic order of the romanization of their Chinese titles which uses the classical conventions for representing Chinese sounds by means of the Western alphabet. These conventions were developed by the oriental scholars Wade and Giles. The information about each serial begins with the item number, that is, the enumeration of the serial in the present list. The Wade-Giles title follows in boldface type. The romanization in the "P'in Yin" conventions developed recently in Mainland China follows in parentheses. The title in Chinese characters is included next. These were not set on the Photon machine, but were stripped in from material set by conventional methods. The English translation of the title, provided by the bibliographic compiler, appears next in square brackets. In some cases a Chinese serial has a title that is spelled in roman letters and included by the publisher on the title page. This is usually a Latin title, for example, *Acta Pharmaceutica Sinica.* When it exists it is included before the compiler's translated title. The imprint, and the bibliographic history if there is any, are given after the translated title. The imprint consists of the place of publication, publisher, date of first publication, if this is known, and the frequency of publication. The abbreviations n.p., n.pub., n.d., and n. freq. are used, respectively, when these are unknown. The bibliographic history states if the serial superseded another, absorbed another, was superseded by another, was absorbed by another, was merged with another, was suspended, or was divided. The individual library holdings appear next. These are listed in the alphabetic order of the abbreviated library names. The holdings of each library are listed chronologically after its abbreviated name. Volume numbers are given first in boldface, then part numbers, and then the publication year. As an illustration, the portion of the *Union List* in Example 6.11 shows that the whole of Volume 7 of the *Chinese Journal of Stomatology* is held by the U.S. Library of Congress and the M.I.T. Library, and that parts 2 to 7 of it are also held by Harvard.

The individual libraries which are covered by the *Union List* sent details of their holdings to the M.I.T. Libraries, where the information was collated and a handwritten card index compiled. Romanized titles were added where necessary, and the source material for the *Union List* was typed by library staff on the two SPD Flexowriters in the C.C.L. The carriage-return key was pressed three times before the material for a serial was started. A new line of Flexowriter copy was started at the beginning of the Wade-Giles title, after the end



83
Chung-hua i shih tsa chih (Zhonghua yishi zazhi)
中华医史杂志

[Magazine of Case History of China]
Peking, People´s Medical Publishers, [1954]. quarterly

DLC	2, 1954

84
Chung-hua k´o hsueh hsieh chin hui: nien hui lun wen chuan chi (Zhonghua kexue xiejinhui: nianhui lunwen zhuanji)
中华科学協進会：年会論文專集

[Chinese Association for the Advancement of Science Annual Meeting Proceedings]
n.p., n.pub., 1955. annual

MiU-A	1, 1, 1955

85
Chung-hua k´ou chiang k´o tsa chih (Zhonghua koujiangke zazhi)
中华口腔科杂志

[Chinese Journal of Stomatology]
Peking, People´s Medical Publishers, [Aug. 1953]. varies,bimonthly

DLC	2, 4, 1954	3-7, 1955-1959
	8, 1-3, 1960	
MCM	3, 2,4, 1955	4-7, 1956-1959
	8, 1-3, 1960	
MH-HY	2, 4, 1954	4-6, 1956-1958
	7, 2-6, 1959	

86
Chung-hua nei k´o tsa chih (Zhonghua neike zazhi)
中华內科杂志

[Chinese Journal of Internal Medicine]
Peking, People´s Medical Publishers, [Jan. 1953]. monthly

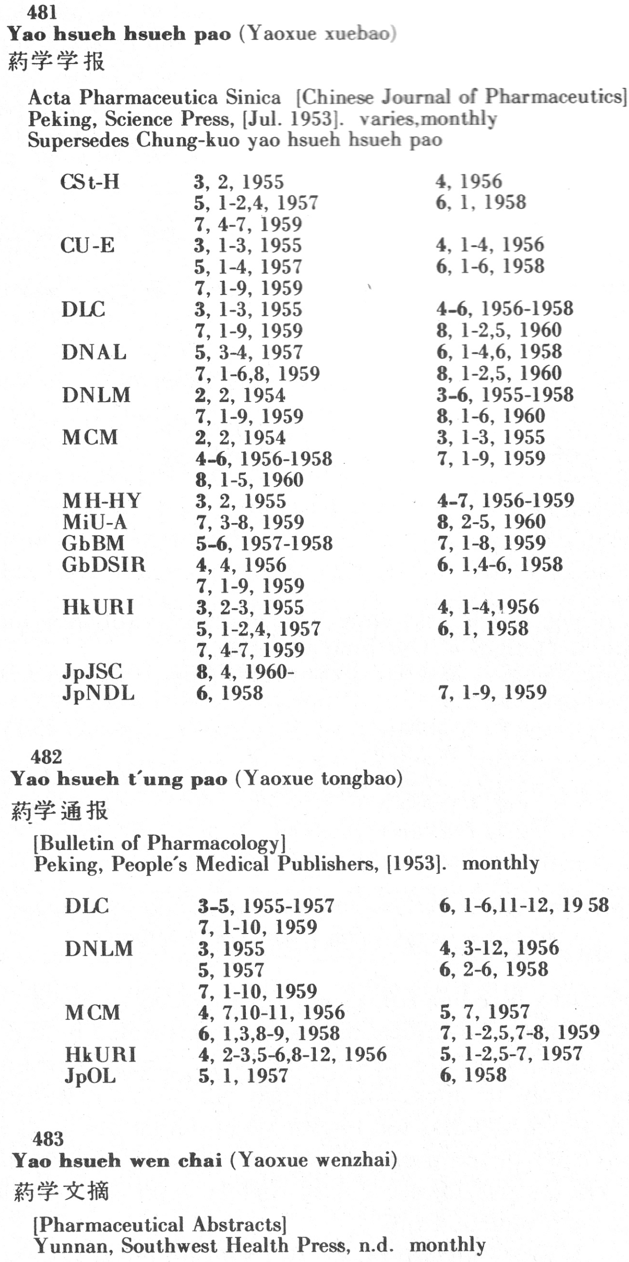
DNLM	2, 1-6, 1954	3-7, 1955-1959
	8, 1-2,5, 1960	
MCM	3, 4-12, 1955	4-7, 1956-1959
	8, 1-2,5, 1960	
MH-HY	2, 6, 1954	4-6, 1956-1958
	7, 1-11, 1959	
GbBM	5, 3-12, 1957	6, 2, 1958
JpOL	6, 1958	

87
Chung-hua p´i fu k´o tsa chih (Zhonghua pifuke zazhi)
中华皮膚科杂志

[Chinese Journal of Dermatology]
Peking, People´s Medical Publishers, [Oct. 1953]. varies,bimonthly

DNLM	2, 4, 1954	3-7, 1955-1959
	8, 1-2, 1960	
MCM	3, 3, 1955	4-7, 1956-1959

Example 6.11. *A portion of the* International Union List of Chinese Communist Serials *produced by the M.I.T. Libraries and the C.C.L.*



481
Yao hsueh hsueh pao (Yaoxue xuebao)
药学学报

Acta Pharmaceutica Sinica [Chinese Journal of Pharmaceutics]
Peking, Science Press, [Jul. 1953]. varies,monthly
Supersedes Chung-kuo yao hsueh hsueh pao

CSt-H	**3**, 2, 1955	**4**, 1956
	5, 1-2,4, 1957	**6**, 1, 1958
	7, 4-7, 1959	
CU-E	**3**, 1-3, 1955	**4**, 1-4, 1956
	5, 1-4, 1957	**6**, 1-6, 1958
	7, 1-9, 1959	
DLC	**3**, 1-3, 1955	**4-6**, 1956-1958
	7, 1-9, 1959	**8**, 1-2,5, 1960
DNAL	**5**, 3-4, 1957	**6**, 1-4,6, 1958
	7, 1-6,8, 1959	**8**, 1-2,5, 1960
DNLM	**2**, 2, 1954	**3-6**, 1955-1958
	7, 1-9, 1959	**8**, 1-6, 1960
MCM	**2**, 2, 1954	**3**, 1-3, 1955
	4-6, 1956-1958	**7**, 1-9, 1959
	8, 1-5, 1960	
MH-HY	**3**, 2, 1955	**4-7**, 1956-1959
MiU-A	**7**, 3-8, 1959	**8**, 2-5, 1960
GbBM	**5-6**, 1957-1958	**7**, 1-8, 1959
GbDSIR	**4**, 4, 1956	**6**, 1,4-6, 1958
	7, 1-9, 1959	
HkURI	**3**, 2-3, 1955	**4**, 1-4,1956
	5, 1-2,4, 1957	**6**, 1, 1958
	7, 4-7, 1959	
JpJSC	**8**, 4, 1960-	
JpNDL	**6**, 1958	**7**, 1-9, 1959

482
Yao hsueh t´ung pao (Yaoxue tongbao)
药学通报

[Bulletin of Pharmacology]
Peking, People´s Medical Publishers, [1953]. monthly

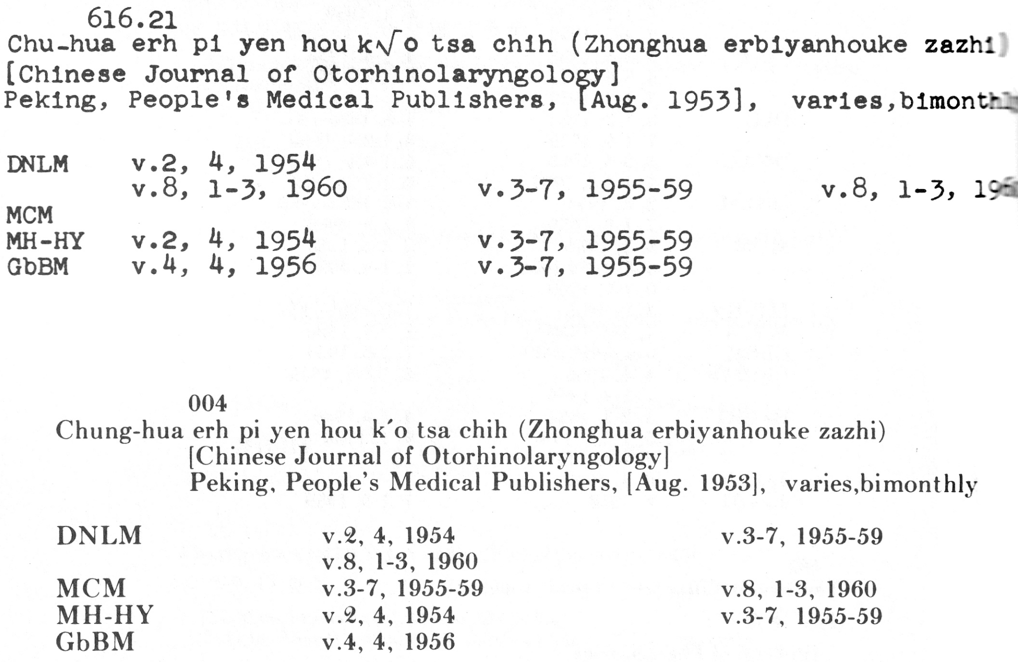
DLC	**3-5**, 1955-1957	**6**, 1-6,11-12, 1958
	7, 1-10, 1959	
DNLM	**3**, 1955	**4**, 3-12, 1956
	5, 1957	**6**, 2-6, 1958
	7, 1-10, 1959	
MCM	**4**, 7,10-11, 1956	**5**, 7, 1957
	6, 1,3,8-9, 1958	**7**, 1-2,5,7-8, 1959
HkURI	**4**, 2-3,5-6,8-12, 1956	**5**, 1-2,5-7, 1957
JpOL	**5**, 1, 1957	**6**, 1958

483
Yao hsueh wen chai (Yaoxue wenzhai)
药学文摘

[Pharmaceutical Abstracts]
Yunnan, Southwest Health Press, n.d. monthly

Example 6.12. *Another portion of the* Union List.

of the P'in Yin title, at the beginning of each item of the bibliographic history, and at the beginning of the holdings information for each library. The carriage was tabulated to different positions on these occasions and before and after the volume numbers which were to be boldface. Copy was otherwise typed continuously and freely without reference to where line breaks might be needed to avoid overset in the typeset output. Diacritical marks were represented by the codes which were punched by the keys labeled with certain mathematical symbols not likely to appear in the text (Σ, √, etc.). Portions of Flexowriter source material and phototypeset output obtained with a prototype version of the program are reproduced as Examples 6.13 and 6.14. The output format differs slightly from that which was adopted in the production run.



```
    616.21
Chu-hua erh pi yen hou k√o tsa chih (Zhonghua erbiyanhouke zazhi
[Chinese Journal of Otorhinolaryngology]
Peking, People's Medical Publishers, [Aug. 1953],  varies,bimonth

DNLM     v.2, 4, 1954
         v.8, 1-3, 1960          v.3-7, 1955-59          v.8, 1-3, 19
MCM
MH-HY    v.2, 4, 1954            v.3-7, 1955-59
GbBM     v.4, 4, 1956            v.3-7, 1955-59
```

```
          004
Chung-hua erh pi yen hou k´o tsa chih (Zhonghua erbiyanhouke zazhi)
          [Chinese Journal of Otorhinolaryngology]
          Peking, People's Medical Publishers, [Aug. 1953],  varies,bimonthly

DNLM            v.2, 4, 1954                 v.3-7, 1955-59
                v.8, 1-3, 1960
MCM             v.3-7, 1955-59               v.8, 1-3, 1960
MH-HY           v.2, 4, 1954                 v.3-7, 1955-59
GbBM            v.4, 4, 1956
```

Examples 6.13 and 6.14. *Flexowriter input and phototypeset output from a prototype version of the* Union List *program.*

Binary cards were punched on the 047 with bit images of the tape codes and written on magnetic tape off-line in the way that is described for TYPRINT input in Section 2.3. These cards served as input to a two-stage computer run. The first stage produced an intermediate representation of the *Union List*, expressed in the ρ-code. In the second stage, the ρ-coded material was converted by part of the PC6 programs into Photon code. The Photon tape was punched off-line and used to operate the Photon machine in the filmsetting process. The film was developed. Numerous errors were found that resulted from misconceptions in the design of the program, misunderstandings of the typing conventions, typographic errors of literal-content, and machine errors (input output equipment and Photon machine). Erroneous material was retyped on the Flexowriter, rerun on the computer and the Photon, and the correct film was stripped in. Chinese characters also were incorporated in the film make-up process, and negatives then prepared for photo-offset reproduction.

The preparation of the *Union List* by computer-aided typesetting was prompted by a wish to mechanize the work of producing the indexes and updating the holdings information for recurrent publication. The production of the main part of the *Union List* took so long, however, that there was no time to write the programs for the mechanized index construction. A manual card-punching and sorting process was used to prepare the indexes. The development of mechanized index production would provide net economies if used for several successive issues of the *Union List*. The experience that was gained in the work that was done will permit more effective planning and, it is hoped, subsequent production of union lists, and the mechanical updating of these. Several aspects of the production of the *Union List* that may be of interest to readers who are concerned with projects of a similar nature are discussed in the further paragraphs of this section.

The columnar format of the *Union List* was used with the object of making the volume compact, while facilitating the location of entries for the journal, library, and year in which a user was interested. Unfortunately, there are numerous cases of one library holding issues of a serial which were published in a single year and which cannot be listed on a single line within the columnar margins that are used. This was not anticipated from the sample on which the design was based. Overset therefore occurred repeatedly when the job was run, and to avoid white space, the film was cut and made up so that the chronological list of holdings sometimes goes down within the left column, and then diagonally upward to the right column. For the same reason, it sometimes goes downward in the right column and then horizontally to the left. A simpler format has been designed for future use which just lists entries from left to right across the entire page. The holdings of one serial in one library require entries in more than two lines in a substantial number of cases in the present format, so that the simpler format which is now proposed should not increase the bulk of the publication by an appreciable extent.

The typing conventions which were used to prepare the input are also subject to several improvements which illustrate a few principles that may be of some generality. These improvements relate to (1) removing certain inadvertent restrictions concerning the length and content of some of the items of the material that is to be set, (2) simplifying the work of the keyboard operators, (3) limiting the amount of output which can be invalidated by an error condition in the computer run, and (4) facilitating repeat runs of material which has been set incorrectly in the first run. They all affect the use that is made of the carriage-return code. It is very convenient, when preparing input to a computer-aided typesetting process, to signal a typographic change by starting a new line of Flexowriter copy that is also characterized by other explicit keyboard actions or by implicit features of literal content. It is unwise, however, to commit the carriage return to the control of output typography to the extent that it always implies a typographic change and cannot therefore be used to start a new line of hard copy when the right margin has been reached on the previous line. This comment may seem too obvious for explicit mention, but it is all too possible for computer staff to design a program with the premise that certain items of text can never occupy more than one line of input and/or output, and with the result that complexity must be piled onto complexity in portions of the program which deal with "switching," when it is found in the course of a production run that the premise is false. It transpired that Wade-Giles and P'in Yin titles, as well as annual holdings, could go on for several lines, even though the sample on which the design was based had only titles which could be accommodated on a single line of Flexowriter copy. It is possible, of course, in situations such as this to use the convention that one or more tabulate actions follow a carriage return which implies a typographic change, and to use the carriage return without a following tabulate when a new line is started because the previous line of hard copy would pass the typing margin.

Different items of information concerning each serial in the *Union List* were set in distinctive ways. In material of a cyclic format, it is possible to distinguish the beginnings of items that are to be treated in typographically distinct fashions by different numbers of consecutive carriage returns. For example, three consecutive carriage returns without intervening characters were used as the signal for the beginning of the material that related to a serial, i.e., the beginning of the Wade-Giles name. In a similar manner, two consecutive carriage returns could separate the P'in Yin from the translated title, and one carriage return could separate the latter from the beginning of the bibliographic history. These conventions could be elaborated to take care of lengthy individual items, say, by using a carriage return followed by single tabulate for overflow input lines. This would enable the computer to distinguish the situations in which there should be a change to boldface (three carriage returns), the provision of vertical space for the Chinese title and indention for the English title (two carriage returns), and the indention of a new line without extra leading (one carriage return). The keyboard actions and blank lines left in the input in this instance would be quite natural in relation to the content of the material that was typed. Further conventions would be needed for the holdings data. A single carriage return would take care of each item of bibliographic history of a serial if there were more than one such item. The disadvantage of using different numbers of adjacent carriage returns is that it can make the hard copy relatively lengthy by requiring a large incidence of blank lines.

It is also possible to design programs which deal in a cyclic manner with material that follows successive carriage returns after some reference point. For example, a set of typing conventions for the *Union List* could use an asterisk * as the delimiter between the material for successive serials in the input, and start a new line for the Wade-Giles title, for the P'in Yin title, for the Latin title (or the translated title if there is no Latin title), and for each item of bibliographic history. Overflow lines could be identified by tabulation, and the holdings information identified by some further convention. It is easy to write a program that uses the detection of an asterisk to start a cycle which puts codes in the output string (1) to start a new line and to switch to boldface when the first carriage return is encountered, (2) to switch to lightface when the next carriage return occurs, (3) to leave three blank lines and to indent the following line when the next carriage return occurs, (4) to provide normal leading and indention on each subsequent carriage return in the input. The work of the keyboard operator is made very simple by this approach, but it has two disadvantages. If the operator makes a mistake by missing out an item of information (say the P'in Yin title) or putting in a redundant carriage return (say by typing two carriage returns between the delimiting asterisk and the Wade-Giles title), the entirety of the material that follows the error in the serial in question is invalidated. "Recovery" after an error thus is precluded until the beginning of the next serial. Furthermore, the input that is needed to rerun a portion of the material that pertains to a serial for stripping in must contain or simulate all the preceding material for the serial in question. For example, if the Western language title is in error, the input to rerun it in the framework of the conventions that are discussed in this paragraph must begin with an asterisk, and then three carriage returns, with or without intervening characters. If it were typed without an asterisk, a program that depended on an asterisk to initiate its basic cycle of operations would probably behave in a meaningless fashion. An asterisk followed by one carriage return would boldface the output. An asterisk followed by two carriage returns would set the output without indention and hence with wrong measure. These problems of error propagation and the need for context simulation in partial reruns beset all uses of precedent material to modify the significance of input codes. Although elaborate cycles can be programmed, they tend to amplify troubles that would be contained by conventions which gave a greater independence to the processing of successive items of input at the expense of some redundancy of coding.

When designing programs to typeset cyclic material, it is expedient to regard certain typographic changes as attributes of the ends of the portions of text which precede them, and other typographic changes as attributes of the portions of the text which follow them. The distinction is artificial if attention is directed to just one isolated change. It becomes very meaningful in relation to the variety of actions which the keyboard operator must peform in an entire job. This sort of consideration can be illustrated by the break point between the P'in Yin title and the Western language title(s) in the *Union List* as it is now set. The indention of the line containing the Western language title(s) in the output can be regarded as an attribute of the beginning of that line. The extra leading that is provided in the Photon run to leave space to strip in the Chinese title in the subsequent film make-up stage can be regarded as an attribute of the end of the line which contains the P'in Yin title. The end of the P'in Yin title can be detected by a program in various ways, for example by the right parenthesis, so that the appropriate additional leading codes may be included in the output when the parenthesis is encountered. The beginning of the first (or only) Western language title then must trigger the provision of codes just for indention, and become typographically equivalent to the beginning of a bibliographic history item. The same convention therefore can be used to start typing the first (or only) Western language title and each bibliographic history item in the input. The convention could be, for example, tabulation to the same position of the carriage. This approach makes the keyboard work simpler than, for example, tabulating to one position at the beginning of the first (or only) Western language title and to another position at the beginning of each bibliographic history item, in order that the number of tabulate codes which precede the Western language title may be used to trigger the inclusion of extra leading (for the Chinese title) and indention codes in the output.

Type face, indention, and measure do not change until some explicit action is taken to alter them. The program need not therefore introduce output codes for these at the beginning of bibliographic history items, since they are established previously by the Western language title(s) line when the complete information for a serial is processed. The disadvantages of making the computer include these codes at the beginning of each bibliographic history item are that extra computer time is required to do so, and extra Photon time is needed to run the tape if it contains redundant disc level codes. The advantage of this redundancy is that it restricts the effect of input and processing errors, and it facilitates the small reruns that may be made to replace erroneous portions of film in the make-up stage.

Although the essence of the production of the *Union List* is that the material which is set conforms to a regular pattern, it is essential in jobs such as this to allow inclusion of unexpected material which does not conform to this pattern. Even though the personnel who plan such a work may be completely certain that they have anticipated and enumerated all the sorts of item that may occur, it can be disastrous to premise the program design upon this "certainty." There must be some provision for items of unanticipated variety. A decision must be made concerning the type face, point size, measure, indention, and leading in which such material will be set, and input conventions established for its inclusion. There must also be some provision for the inclusion of blank lines in the output at positions which are specified in the input and which are not part of the standard repetitive output format. Even if all the items that can result from the indexing of existing works have been enumerated, no bibliographer can anticipate or control the quirks and absurdities of future producers of printed publications, particularly with the prospect of widespread in-house computer-aided typesetting in view.

Blank lines merit a little comment. At times, it is quite useful to include blank lines in the input which have no effect on the output.

It may be expedient to leave space between particular lines for clarity in proofreading or for the addition of written comments at points which have no typographic significance. Blank lines may be left by pressing the carriage-return key arbitrarily at the beginning and end of pages of hard copy and at points in the text where there is a substantial break of substance. Reasons thus exist for enabling typesetting programs to ignore blank lines of input, that is, carriage returns which are adjacent or which are separated by only space codes. On the other hand, one of the ways of distinguishing break points at which different typographic actions must be taken is by using differing numbers of adjacent carriage returns. If typography is controlled by carriage returns, some further code convention is needed to distinguish blank lines (i.e., carriage returns) which are typographically significant from those which are not. It is possible to devise input conventions that treat a carriage return as significant only if it is (or is not) followed by some other code, such as a tabulate code, but elaborating the conventions confuses the operator and increases the possibility of errors. Moreover, conventions that leave no record on the hard copy are to be avoided if possible, and tabulates and spaces galore can appear or be omitted between carriage returns without any visible evidence until erroneously set film has been developed. In practice, a compromise must be struck between the need for flexibility and "open endedness" on the one hand and for simplicity of operation and ease of checking on the other. This discussion is intended, not to suggest that the design of conventions and programs for a job such as the *Union List* is subject to irreconcilable considerations, but rather to stress that the consequences of a set of input conventions can have many ramifications throughout a job. It is always easy to be clever after the event; the problem is leaving a buffer for the lack of omniscience beforehand.

The *Union List* production is being redesigned, and some features of a possible revised procedure may be mentioned briefly. The output and input formats have been simplified to the extent that only three different end-of-line actions need be performed by the keyboard operator. It has been contrived that in each case there are three key strokes, either (1) carriage return, space, space; (2) carriage return, tabulate, space; or (3) carriage return, tabulate, tabulate. This use of three key strokes in every case was suggested by consideration of the conversion of the Flexowriter tape to Hollerith cards of a convenient format. It seems that the regular rhythm which it would allow the operators at the end of each line would facilitate their work. The proposed use of Hollerith cards as the intermediate between the Flexowriter tape and the computer run devolves from a consideration of error correction procedures. Perhaps the weakest part of the *Union List* production was the dearth of editing facilities. The considerations that are discussed in Section 6.7 in connection with the "Picture Book" application are pertinent to the design of effective error correction procedures for a job such as the *Union List.*

6.6 The *Current Serials and Journals* Catalog

Numerous catalogs containing names and other words are punched on decks of Hollerith cards that can be updated easily and used to print successive editions on conventional punched-card equipment. The catalogs would be more readable and would require less paper if they were set in type, but in many cases it would be far too costly and time consuming to do this by keyboarding the material again for a typesetting machine. It may be possible in these cases, however, to use a computer to produce a tape for a mechanically operated typesetting machine, from the decks of Hollerith cards that are punched with the catalog entries.

In general, catalogs that have been maintained on punched cards, and which were started before the advent of computer-aided typesetting or without the possible use of this technique in mind, do not distinguish upper- and lower-case letters. The capability of the typesetting machine to set letters in both cases and in different fonts cannot be utilized unless the computer determines which letters should be capitalized and where the font should be changed. The TABPRINT option of the PC6 system enabled material in different card fields to be set in different fonts, provided that the same font was used for material in the same card field throughout the deck. The TABPRINT option could not provide changes of font at positions that varied from card to card. Such changes are needed, for example, when a person's name and address are punched continuously from left to right across a card, and only the name is to be boldface in the typeset end product. To deal with this situation and with the capitalization of appropriate letters, some further programs are needed to take into account the rules that determine where case and font should be changed. Programs of this type were written for two catalogs that are of interest at M.I.T.: *Current Serials and Journals of the M.I.T. Libraries* and the *Student Directory*. The operation of the programs was demonstrated by the production of operating tapes to set a few pages of each of these catalogs on the C.C.L. Photon machine.

Current Serials and Journals lists the names and holdings of all the periodical publications which have been received by the M.I.T. Libraries. It has been maintained on punched cards for several years. It is updated and reprinted annually by listing the cards in an IBM 407 tabulator on heavy paper and reproducing the product by photolithography. Each periodical title is accompanied by the volume or part number and date of the first issue that is held by any of the libraries at M.I.T., followed by a plus symbol if it has been taken without interruption since then. Variations on this style of recording the holdings are used when the acquisition of the periodical has been interrupted or terminated and when the periodical has changed its title or combined with another periodical. The name of the library in which the periodical is kept is given next, followed by the call mark which locates the periodical in the library. If copies of the periodical are kept in more than one library at M.I.T., the names of all the relevant libraries are given. Variant names of periodicals are included in the catalog, which is arranged alphabetically, and cross references are provided to the version of the name which is accompanied by the holdings and location data. A portion of the catalog that was printed on an IBM 407 is reproduced as Example 6.15.

The PC6 system was supplemented by some further programs, that are described in Section 7.5, to typeset *Current Serials and Journals* from the card deck in which it is punched in Hollerith code. Information concerning a single periodical is punched in one or more cards. The title starts on column 1 of the first (or only) card, and if it requires more than 50 card columns it is continued in columns 3 to 50 of one or more continuation cards. The holdings data are punched in columns 53 to 60, the library name or names in columns 61 to 70, and the call mark in columns 71 to 80. The holdings information may appear on more than one card (but only in columns 53 to 60). The library name(s) and call mark for a single periodical rarely extend beyond one card.

For the production of the typeset version, the first letter of each complete word, with the exception of certain connective words, was capitalized. Also capitalized were the first letter of each portion of a hyphenated word and the first (or only) letter of an abbreviation that was followed by a period. Only the first letter of an acronym was capitalized, so that, for example, the abbreviation for *Allmanna Svenska Elektriska Aktiebolaget* was set as A.S.E.A., but the acronym was set as Asea. The rules for capitalization in the title field amounted to capitalizing every letter that either started a title, followed a blank and did not start one of the words in a reference list of exceptional words, or followed a period, comma, hypen, slash, or left or right

```
A.E.G. PROGRESS
    (ALLGEMEINE ELEKTRICITATSGESELLSCHAFT, BERLIN)
NO.1,APRIL 1925-1934              ENG G     VAIL
1956,NO.1+
A.E.I. ENGINEERING
     (ASSOCIATED ELECTRICAL INDUSTRIES, LTD.)
V.1,1961+                         ENG G     VAIL
A.I.B.S. BULLETIN
     (AMERICAN INSTITUTE OF BIOLOGICAL SCIENCES)
V.1,1951+                         SCIENCE   BIOL P 570.51 A51
A.I.CH.E. JOURNAL
     (AMERICAN INSTITUTE OF CHEMICAL ENGINEERS)
V.1,NO.1,MAR 1955+                SCIENCE   660.51 A11
```

Example 6.15. *A portion of the catalog,* Current Serials and Journals of the M.I.T. Libraries, *as printed on an IBM 407 tabulator.*

TITLE	HOLDINGS	LIBRARY	CALL NUMBER
A.E.G. Progress (Allgemeine Elektricitatsgesellschaft, Berlin)	no.1,April 1925-1934 1956,no.1+	ENG G	Vail
A.E.I. Engineering (Associated Electrical Industries, Ltd.)	v.1,1961+	ENG G	Vail
A.I.B.S. Bulletin (American Institute of Biological Sciences)	v.1,1951+	SCIENCE	Biol P 570.51 A51
A.I.Ch.E. Journal (American Institute of Chemical Engineers)	v.1,no.1,Mar 1955+	SCIENCE	660.51 A11
A.I.P. Documentation Newsletter (American Institute of Physics)	v.1,no.1,July 1959+	*HAYDEN	P 530.51 A11
A.O.P.A. Pilot (Aircraft Owners and Pilots Association)	v.2,no.8,Aug 1959+	AERO	
A.P.C.A. Abstracts (Air Pollution Control Association)	v.2,n .4,Sept, no.6,Nov 1956+	ENG G	

Example 6.16. *A portion of the catalog, as printed by computer-aided typesetting.*

parenthesis. The list of exceptional words was compiled by inspection of the most recent issue of the catalog. It comprised the following words: throughout, following, divided, parts, entries, issued, before, under, bound, thru, only, also, with, into, och, aus, van, etc., per, die, des, und, der, zur, fur, and, for, the, of, in, de, do, es, et, le, la, on, di, si, po, ed, ee, du, by, to, i, l, v, y, e, s, d, a. The entire contents of the title field was boldface unless the field started with an abbreviation, in which case this alone was boldface, or unless the field contained parenthesized material, in which case only the material preceding this was boldface.

In the holdings field words were capitalized by reference to a list which had been compiled by inspection of the last issue of the catalog, and which contained the days of the week, abbreviations for the months of the year, and the following words: meeting, current, argang., spring, autumn, winter, summer, issue, jahrg., annae, anul, fall, lief., vol., ser., new, bd., ano, special. The library names were capitalized in full. In the call mark, the first letter was capitalized, and so also was any later letter which was preceded by a blank.

The program produced material for pages of specified dimensions and headed the successive columns TITLE, HOLDINGS, LIBRARY, and CALL NUMBER. The first two letters of the first title on each page was printed at the top right-hand corner. Running time was dominated by the capitalization process. A portion of the typeset material is reproduced as Example 6.16. It was not possible to run the job in full before the termination of the typesetting project at M.I.T.

The *Student Directory* application dealt with a punched-card file of names, addresses, and telephone extensions of the members of the undergraduate school. Some simple rules for capitalizing initials and first letters of names were defined, and a program that corresponded to these was written by methods similar to those which were used for the *Current Serials and Journals* catalog and which are described in Section 7.5. The program was demonstrated, but the job was not run.

6.7 The "Picture Book" Application

In all the applications of the PC6 system that have been discussed so far in this book, the problems of producing just single editions of the respective publications dominated the design and operation of the computer programs. Updating procedures, where they existed, were minimal, and they were used only to correct errors. The "Picture Book" application was undertaken as an experiment by Mr. G. F.

Clancy to demonstrate an approach that was thought to be of possible value for typeset catalogs which undergo periodic revision and reissue.

The "Picture Book" which formed the subject of this study is *Members of the Faculty and Administration*, an annual M.I.T. publication. It contains the photograph of each member of the Faculty and the Administration. The name, titles, and degrees of each staff member appear beside the member's photograph. The text information concerning each staff member consists of some 60 to 100 characters set on three or four successive lines. These appear beside the photograph to which they apply. About 30 per cent of the entries in one issue differ from those in the preceding issue because of arrivals, changes of title, and departures.

A single edition of the "Picture Book" could be produced by typing the information about each staff member in turn on the Flexowriter in the TYPRINT conventions and using the PC6 programs to translate these into Photon codes, as described in Chapter 3. An updated edition could then be produced by typing a list of changes on the Flexowriter as described in Section 3.16, and using the computer first to update the representation of the entire "Picture Book" in Flexowriter code and then to translate this in its entirety into Photon operating code.

When preparing a catalog, it is of some convenience to record the successive entries on cards that can be arranged as a card index. In particular when dealing with catalog information that is to be processed by a computer, it is convenient to punch the successive items in Hollerith cards that are "interpreted," that is, the contents of the cards are printed near their upper edge. Changes of one or two items in the catalog then can be made manually. Auxiliary punched-card machines, or the computer, need be used to update the card deck only when the number of changes makes mechanical updating more convenient than manual revision. The information for the "Picture Book" could therefore be typed on a Flexowriter, and Hollerith cards punched from the tape by an 047 punch and passed through an interpreter to print onto them the information that had been punched. The 047 control panel could be wired so that some simple typing convention caused the 047 to start a new card for each staff member and so that the cards were punched in the BCDPRINT conventions or in some slight variant of these. Alternatively, the source material could be typed on a card punch in the BCDPRINT conventions directly. The card deck could then be processed using a program with the capabilities of the BCDPRINT option of the PC6 system, to produce one edition of the "Picture Book." To produce a subsequent edition, changes would be typed on the Flexowriter and converted to cards, or typed on cards directly, and the card deck updated manually or on a collator, or an updated deck could be produced by use of a simple computer program. The updated deck then would be processed to produce the new edition in the same manner as the original deck had been processed to produce the original edition.

The approach which has just been described would have the advantage of allowing manual revision of the card deck that was punched with the information that was to be typeset. A similar approach could be used with straight text matter by punching Hollerith cards in the BCDPRINT conventions (or a slight variation of these) from the Flexowriter tape and processing these cards through the interpreter. Changes could be made by repunching individual Hollerith cards on a card punch, or by punching tape on a Flexowriter and then punching Hollerith cards from this on an 047 and updating the deck manually for small changes and mechanically for larger changes. The author is not sure of the extent to which this might be advantageous for straight text matter, but it certainly merits investigation. In any case, the use of binary cards as the intermediary between Flexowriter tape and the computer was a nuisance, for it involved the operation of the 047 in a marginal fashion for which it is not designed.

It should be mentioned that the IBM 870 document writer enables cards to be punched from a typewriter keyboard, with case shift indication, and hard copy typed with upper- and lower-case letters at the same time, or in a subsequent operation. This equipment is of considerable usefulness in situations that require a card file of material which is to be printed in upper and lower case.

To return to the subject of the "Picture Book," processing a card deck with the BCDPRINT option of the PC6 system or some similar program, as described in this section, would provide the benefit of allowing manual revision of the source material when appropriate, but it would require complete code translation of the entire "Picture Book" from Hollerith code to Photon code for each edition. Now changes in a catalog such as the "Picture Book" tend to have a much more localized effect in the operating tape for the typesetting machine than do changes in text material that consists of sentences and paragraphs. Changing a single word can alter the positions of line breaks and interword spaces throughout a paragraph of straight text matter. In the "Picture Book," however, the positions of line breaks are unchanged in the entries for staff members who have stayed without change of title since the time of the previous edition and often for those whose titles have been changed. In many catalogs each entry occupies one line, so that changes in one entry cannot affect the positions of other line breaks. Catalogs therefore present much greater possibilities than straight text matter for eliminating redundant code translation in the production of revised editions. It is possible when revising catalogs to translate just the new and altered entries into operating code and to merge the resulting sequences of operating codes for these with the sequences of operating codes for the unchanged entries in the complete sequence of operating codes from which the previous edition was set.

It is of some convenience, for the process which has just been described, to record the *operating* codes on a card deck in which individual cards or groups of successive cards correspond to individual catalog entries, and in which the card or cards for a given entry can be located easily. In particular, it becomes possible to make small changes to the deck manually by inserting and/or removing cards punched with Photon codes, if these cards also are punched in part, or preceded by, cards that are punched with Hollerith codes that have been interpreted for visual identification. A modified form of the PC6 system was used to process material for the "Picture Book," as described in the next paragraph, to demonstrate this approach to catalog setting. Production runs could not be attempted before the termination of the project at M.I.T.

"Picture Book" source material was keyboarded on a Hollerith card punch. The dollar symbol was used as a case shift indicator. The input deck for the production of "first edition" is illustrated by the short test deck whose listing is reproduced as Example 6.17. A new card was started wherever a new line of output was required by the content of the material that was being typed. This occurred at the beginning of a staff member's name, at the beginning of the member's title (or first title if there were several), and at the beginning of the list of degrees. A blank card was left between the last card for one staff member and the first card for the next.

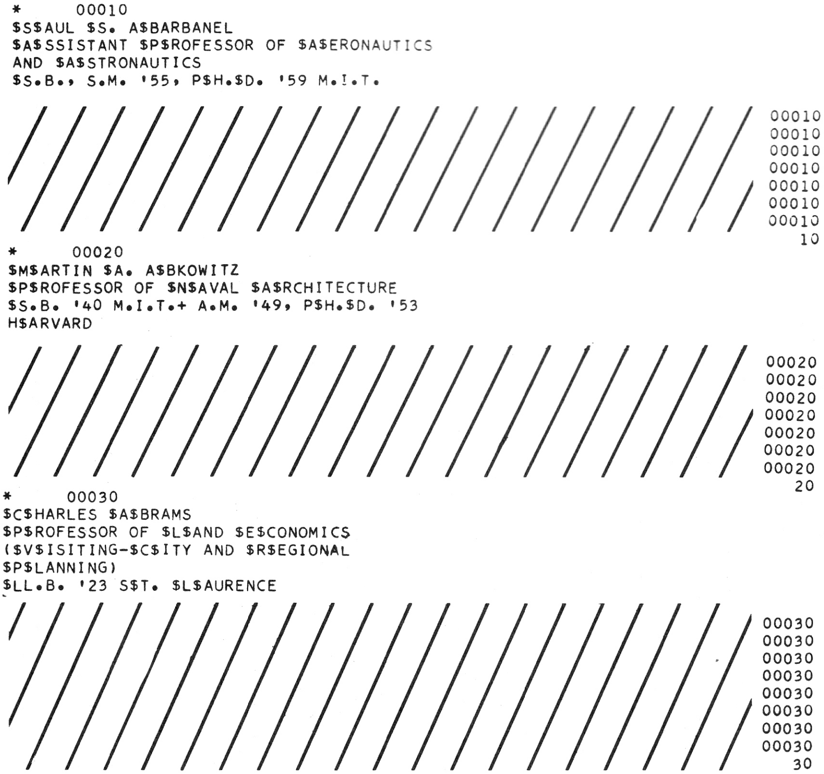
For the production of a first edition, the input deck consisted of (1) a card punched * MASTER, (2) a card punched *78 to select the escapement table for disc 78, (3) three cards punched with groups of control codes that were obeyed cyclically in the processing of material for each staff member, (4) the card groups for the staff members, arranged alphabetically, (5) a card punched * END. The modified PC6 programs produced an output deck whose structure is illustrated by the listing (Example 6.18) of the cards which were produced from the input deck listed in Example 6.17. The output

```
*      MASTER
*78
*IN,LD2,BD10,DL3,LS9,FL*
    *NL,DL2*
*VP132,EN*
$S$AUL $S. A$BARBANEL
$A$SSISTANT $P$ROFESSOR OF $A$ERONAUTICS
AND $A$STRONAUTICS
$S.B., S.M. '55, P$H.$D. '59 M.I.T.

$M$ARTIN $A. A$BKOWITZ
$P$ROFESSOR OF $N$AVAL $A$RCHITECTURE
$S.B. '40 M.I.T.+ A.M. '49, P$H.$D. '53
H$ARVARD

$C$HARLES $A$BRAMS
$P$ROFESSOR OF $L$AND $E$CONOMICS
($V$ISITING-$C$ITY AND $R$EGIONAL
$P$LANNING)
$LL.B. '23 S$T. $L$AURENCE

*END
```

Example 6.17. *Input test deck for a first edition of the M.I.T. "Picture Book."*

deck consisted in general of a succession of groups of cards. One group was punched for each staff member. The order of the staff members in the input was preserved in the output. Each output group consisted of (1) a card punched with an asterisk * in column 1 for identification and a reference number in columns 7 to 11; (2) a replica of the input cards for the staff member; and (3) several cards punched in column binary with the Photon codes for that staff member in columns 1 to 72 and the reference number in Hollerith code in columns 73 to 77. The binary punches are not shown in the example. The reference number was ten times the ordinal number of the entry in the "Picture Book." It was constructed automatically by the computer. The first edition output deck was passed through the interpreter which printed just the Hollerith information in the successive cards along their upper edges. The cards punched with Photon codes were immediately identifiable by the reference numbers interpreted from columns 73 to 77. A Photon tape for the first edition was punched from the first edition output deck by a process which ignored the Hollerith cards, and was used to set the first edition. The output cards were also kept for subsequent updating.

The structure of the input deck that was used to produce a revised edition is illustrated by the test example whose listing is reproduced as Example 6.19. The deck consisted of (1) a card punched * EDIT; (2) a card punched *78 as in the first edition deck; (3) three cards that were the same as the cards in the same position of the first



```
*      00010
$S$AUL $S. A$BARBANEL
$A$SSISTANT $P$ROFESSOR OF $A$ERONAUTICS
AND $A$STRONAUTICS
$S.B., S.M. '55, P$H.$D. '59 M.I.T.
```

```
00010
00010
00010
00010
00010
00010
00010
   10
```

```
*      00020
$M$ARTIN $A. A$BKOWITZ
$P$ROFESSOR OF $N$AVAL $A$RCHITECTURE
$S.B. '40 M.I.T.+ A.M. '49, P$H.$D. '53
H$ARVARD
```

```
00020
00020
00020
00020
00020
00020
00020
   20
```

```
*      00030
$C$HARLES $A$BRAMS
$P$ROFESSOR OF $L$AND $E$CONOMICS
($V$ISITING-$C$ITY AND $R$EGIONAL
$P$LANNING)
$LL.B. '23 S$T. $L$AURENCE
```

```
00030
00030
00030
00030
00030
00030
00030
00030
   30
```

Example 6.18. *Output deck structure for the "Picture Book" (cross-hatching represents binary Photon codes).*

```
*      EDIT
*78
*IN,LD2,BD10,DL3,LS9,FL*
    *NL,DL2*
*VP132,EN*
*      00010 DELETE

*      00021 INSERT
$C$LYDE $M. A$DAMS, $J$R.
$P$ROFESSOR OF $M$ETALLURGY
$S.B. '49, S.M. '50, S$C.$D. '53 M.I.T.

*      00030 CHANGE
$D$OUGLAS $P. A$DAMS
$A$SSOCIATE $P$ROFESSOR OF $E$NGINEERING
$G$RAPHICS ($M$ECHANICAL $E$NGINEERING)
$S.B. '30, A.M. '30 H$ARVARD

*END
```

Example 6.19. *Test input for revised edition of the "Picture Book."*

edition deck; (4) cards for each entry in the Picture Book that was to be deleted, changed, or added; and (5) a card punched * END. To delete an entry, a single card was punched with an asterisk * in column 1, the reference number of the entry to be deleted in columns 7 to 11, and the word DELETE in columns 13 to 18. The reference number that was used was the number which had been allocated by the computer in the production of the previous issue. To insert an entry, the user of the program allocated to the entry a reference number that lay between the reference numbers of the entries between which it was to appear. A card was punched with an asterisk in column 1, this reference number in columns 7 to 11, and the word INSERT in columns 13 to 17. Three or four cards then followed on which the material to be typeset was punched in Hollerith code in the conventions that are described above for entries in the first edition. To change an entry, a card was punched with an asterisk in column 1, the reference number in columns 7 to 11, and the word CHANGE in columns 13 to 18. The new form of the entry was punched in its entirety in the three or four cards that followed.

The modified PC6 programs produced, from this "revision" input deck, an output deck that consisted of a succession of card groups. Each output group started with a copy of the Hollerith card or cards that had been provided in the input to delete or to change or to insert an entry in the Picture Book. The Hollerith cards for changes and insertions were followed by cards that were punched with the appropriate Photon codes in column binary in columns 1 to 72, and with the reference numbers in Hollerith code in columns 73 to 77, in just the same way as the first edition output cards. The revision cards were also interpreted in the manner already described for the first edition output cards. Since first edition and revision output cards could be identified visually, it was possible to update the first edition deck manually when the number of changes was so small that this was easier than submitting a computer run to perform the merging. A simple program was written to perform this merging, for use when the number of changes made manual updating tedious. The updated deck that was produced by either method then could be used to punch a Photon tape to set a revised edition and also could be kept to be updated again.

The procedure could have been varied in several ways, for example by making the computer punch just a single Hollerith card with the staff member's name at the beginning of each output group. The precise method that was used in this study allowed particularly simple modifications of the PC6 programs to suffice, and it is reported here only to indicate the sort of procedure that can be adopted in such catalog work. The details of such procedures will depend on the details of the jobs to which they must apply and will be specified most aptly by personnel who have experience of the jobs, rather than by copying an elaborate prototype that was designed for a job with significant differences.

6.8 Setting Built-up Formulas

Computers are being used to an increasing extent to perform tedious algebra, as explained in Section 6.3. This is giving rise to a need for programs to produce operating tapes to set the built-up formulas on mechanically operated typesetting equipment, from the sorts of representation of the formulas that are produced as the primary results of the programs which do the algebra. The manual keyboard methods that are used at present to set built-up formulas, moreover, involve a considerable amount of routine effort on the part of the keyboard operator to determine the positions in which the characters must be set, going from left to right, line by line down the page. The order in which characters are set in this process often differs considerably from the order in which they are written by a mathematician or generated by a computer program which is performing algebraic operations. The determination of the positions and sizes of the typeset characters and of the order in which they are set is completely routine. There may be scope for using a computer to produce an operating tape for a built-up formula from an input tape that has been punched by a keyboard operator whose work is simpler than at present.

It seems to be of some interest in connection with these matters to experiment with ways of representing a built-up formula by a mixture of characters that are to be set in type and characters that form control codes to define the relative positions in which the typeset characters should be set. A sequence of characters which represents a built-up formula in this way and which can be typed in a straight line is called a linearized representation of the formula in the discussion that follows.

The technical problems of programming a computer to deal with the punched cards or paper tapes that are produced by typing such a linearized representation of a built-up formula on a keyboard perforator are quite considerable. To get some insight into these programming problems, it seemed expedient to begin the study by punching the linearized representations in Hollerith cards on an

IBM key punch. The small character set of the card punch and its inability to produce hard copy were disadvantages, but these were outweighed for exploratory purposes by the advantage of being able to use some fairly elaborate programs which had been written for other purposes and which required input that was expressed in Hollerith code. The conventions that were developed to represent formulas in a linearized style are rather unwieldy, but it should be noted that these correspond to an interim stage in the evolution of a method which probably can be refined to be much more practicable.

Input for the experimental system that was developed by J. M. Gerard[18] for use with the C.C.L. Photon was typed on a standard IBM 026 card punch. The digits 0 to 9 and the characters

+ - / , . and =

were typed as such for inclusion in a typeset formula. The letters A to Z were typed to produce the corresponding lower-case letters in the typeset end product. To set a letter in upper case it was typed between asterisks. Thus *B*ER in the input produced Ber in the typeset film. Greek letters and further mathematical symbols were represented by mnemonics that were enclosed between asterisks. The mnemonic for a lower-case Greek letter contained the letters GL and a third letter whose phonetic value corresponded roughly to that of the Greek letter concerned. Thus *GLA*, *GLB*, and *GLG* stood for α, β, and γ, respectively. The mnemonic for an upper-case Greek letter contained U instead of L as its middle character but otherwise was the same as the mnemonic for the lower-case letter. Thus *GUG* and *GUL* stood for Γ and Λ. The mnemonics for the further mathematical symbols were fairly obvious, e.g. *DIV*, *MULT*, *GRTR*, and *LESS* for $\div$, $\times$, $>$, and $<$, respectively.

A sequence of characters and mnemonics produced the corresponding sequence of typeset characters in a uniform size and in a straight line. Thus *GUG**LBA* -1/2*RBA* produced $\Gamma(-1/2)$. The mnemonics *LBA* and *RBA* were used to set parentheses because the Hollerith brackets were needed for other purposes that are described below. When only a single size of type was needed, the program automatically provided 10-point type.

Material was set on two or more lines by the use of two-letter control codes that were enclosed between dollar symbols. Each control code specified the position of the material that followed it, relative to the position of the material that preceded it. Thus the code RU stood for right upper, so that X$RU$N caused the letter x to be set with the letter n by its right upper corner. The letter x was set in

10-point type as it came first in the input, and the letter n was set in 6-point type as it followed the RU code. The output therefore appeared as x^n. The control codes RL, LU, and LL stood for right lower, left upper, and left lower, respectively, so that X$RL$N, X$LU$N, N$RU$X, and N$RL$X produced x_n, ${}^n x$, n^x, and n_x, respectively. In all these cases, the material before the subscripting or superscripting control code was set in 10-point and the subscript or superscript in 6-point.

A subscript or superscript containing several characters was enclosed in Hollerith parentheses in the input. Thus XRL(I,J) produced $x_{i,\,j}$. Subscripts and superscripts could be attached at more than one corner of a character, for example *C*LUNRLR produced ${}^n C_r$, and *P*RUMRLN$*LBA*COS*GLT*RBA* produced $P_n^m(\cos\theta)$. Subscripts and superscripts could be placed on subscripts and superscripts by a continued use of the conventions which have been described. Thus (XRLI)RLJ produced x_{i_j} with the x in 10-point and the i and j in 6-point.

Material also could be arranged vertically by use of the CU and CL control codes which stood for center upper and center lower, respectively. Thus *X*CU*ARRW* produced X with the character X in 10-point type and the arrow in 6-point type, $\vec{X}$.

Two further control codes LV and RV were used to vary the size of characters. Their use is illustrated by the input

(XRU2)LV*LBA*RV*RBA*

which produced $\left(x^2\right)$, with the character x in 10-point type, the character 2 in 6-point type and the parentheses increased to 14 points. The control code LV applied to the material which followed (either a single character or a number of characters within parentheses), and specified that this be set on the left of the material that preceded the control code and expanded in size to be commensurate with it in the vertical direction. The control code RV set the material that followed it to the right of the material that preceded it, again expanded in size to be commensurate vertically with the material that it follows.

The program in effect treated each character as if it occupied a rectangle and adjoined the rectangles for a sequence of characters that were to be set in a uniform size and in a line. The rectangle for a superscript was placed so that its lower edge was 2 points below the upper edge of the rectangle to which it was appended. The rectangle for a subscript was placed so that its upper edge was 2 points above the lower edge of the rectangle to which it was ap-

$$F(t) = 1/2\pi i \int_{a-i\infty}^{a+i\infty} e^{st} f(s)ds \qquad \int_a^b u.d\dot{v} = [u.v]_a^b - \int_a^b v.du$$

$$(x+y)^n = \sum_{r=1}^{n} {}^nC_r.x^r y^{n-r} \qquad \Gamma(z) = \int_0^\infty e^{-t} t^{z-1} dt$$

$$\{\vec{A} \times \{\vec{B} \times \vec{C}\}\} = \vec{B}(\vec{A}.\vec{C}) - \vec{C}(\vec{A}.\vec{B})$$

$$E(k,\phi) = \int_0^\phi [1-k^2 \sin^2\phi]^{1/2}.d\phi \qquad (\phi = \sin^{-1} k)$$

$$f(x+\Delta x) = f(x) + \sum_{r=1}^{p-1}\{f^{(r)}(x).(\Delta x)^r/r!\} + f^{(p)}(x+\theta.\Delta x).(\Delta x)^p/p! \qquad (0<\theta<1)$$

$$F(k,\phi) = \int_0^\phi [1-k^2 \sin^2\phi]^{-1/2}.d\phi \qquad (\phi = \sin^{-1} k)$$

Example 6.20. *Some formulas phototypeset with an experimental program by J. M. Gerard.*[18]

pended. Gerard's program made the left (or right) vertical edge of a subscript (or superscript) rectangle coincide with the right (or left) edge of the rectangle which it appended. This did not allow "kerning," that is overlapping the rectangles, as is necessary, for example, when setting the limits on a sloping integral symbol.

Some examples of formulas set by Gerard's program are given as Example 6.20. It may be possible to extend the programming procedure which he developed to deal with linearized representations of formulas to increase the convenience of these, perhaps by using a keyboard perforator with a larger character set. An alternative approach to simplified keyboard work for built-up formulas that has been developed elsewhere is mentioned in Section 8.9.

Gerard's work used an "infix" notation (i.e., interspersed control codes) to represent the relative positions of items to be set in type. Another possible method for representing built-up formulas would be by means of a "prefix" notation. For example, certain characters might be reserved as "form specifiers" for particular "forms," and other characters reserved as delimiters to identify text characters and character sequences that followed the form specifiers. Thus a notation of the form $\phi(a,b,c)$, where a,b,c stood for text characters or sequences of text characters, could be used for a portion of a formula in which the character(s) denoted by a were set in 10-point, with the characters denoted by b and c centered vertically below and above it in 6-point. Then $\phi(\Sigma,\mathrm{i}=1,n)$ and $\phi(\Pi,\mathrm{i} > \mathrm{j},)$ would denote, respectively,

$$\sum_{i=1}^{n} \text{ and } \prod_{i>j}$$

7. Programming Considerations

7.1 Introduction

The discussion in this monograph has tended so far to avoid matters of computer programming. Editorial personnel who use the programs such as those that formed the PC6 system need not have any understanding of these matters. The general nature of computer programming, however, is quite comprehensible to people without mathematical or scientific training, and the author has found that users of varied backgrounds have sought information about the general design of the programs from general interest. As mentioned in Section 4.2, moreover, such users have been able, from an understanding of this information, to contribute helpful suggestions concerning simple modifications that would improve the utility of typesetting programs.

Some of the basic concepts of programming, including the use of compilers and of subprograms, are presented in Sections 7.2 and 7.3. The general characteristics of the preprocessors of the types that were used in the PC6 system and which might well be used in future typesetting programs are discussed in Section 7.4. The programming technique of mechanized syntactic analysis that is particularly useful in computer applications that deal with text matter is discussed in Sections 7.5 and 7.6 by reference to a typesetting problem and to an editing program. The fragmentation of a typesetting program into subprograms that deal with logically separable subprocesses is illustrated further in Section 7.7 by a brief description of the ρ-ω translator of the PC6 system. Some features that would be useful in future typesetting programs are discussed in Sections 7.8 to 7.10.

7.2 Input, Storage, and Output

Computers and programming have been the subject of so many popular reports that there is by now quite widespread awareness of the general nature of automatic data processing and of the operation of stored programs. Since computers were designed originally to do numerical work, accounts of programming have tended to present matters by reference to mechanized arithmetic. Computers are also being used increasingly to deal with text material. The computer programs that produced the operating tapes for the Photon machine from TYPRINT and TABPRINT input come largely in this latter category. The emphasis in such work is on the selection, rearrangement, and translation of codes that are read mechanically from an input tape into codes that are recorded mechanically on an output tape. Codes that are punched in cards or paper tapes and represent letters, digits, and other symbols are read mechanically by a computer which then behaves in a way that is determined by the codes which it has read. In this general respect, a computer may be compared with a tape-operated typesetting machine, though the work that computers and typesetting machines do are very different.

When a computer reads information mechanically, it records this in a store. The sequence of output codes is constructed from the stored input by means of rules that are expressed in coded form and recorded in the store before the input data are read and processed. The program of a computer consists of these rules, in the form of coded instructions. The computer store consists of a very large assembly of units called storage locations, which can each record any one of a considerable number of different codes. In the main store of the IBM 709 computer there are over 32,000 storage locations which can each record any one of 2^{36} different codes. The storage locations are numbered consecutively for purposes of identification, and the positional number of a storage location within the entire store is called its address.

It is customary when using large computers to copy the input for a succession of runs from the primary input medium such as punched paper tape or cards onto a single magnetic tape, on off-line equipment, and then to transfer the magnetic tape to a unit that is called a tape drive and which is attached to the computer. The batched jobs are then run consecutively under the automatic control of a "monitor program," for reasons of efficiency. When a very long program must be used repeatedly, it is irksome to rewrite it from a card deck onto the batched input tape every time the program is used. In this circumstance, it is customary to keep the long program on a magnetic tape that is mounted on one tape drive of the computer when the program is to be used. The data for jobs that use the program are written on the input tape that batches these data with the other jobs that are to be processed without interruption under the control of the monitor program. This was done with the PC6 typesetting programs when they were used on a routine basis. Large disc stores are now attached to many computers and programs that are required fairly often may be retained permanently in these to avoid the need to reload a program tape.

Text processing programs such as the PC6 system deal largely with sequences of codes that are stored in consecutive storage locations. In computing terminology, a sequence of symbols or codes is called a string, and a program that constructs one string from another is often called a string transforming program. Such programs depend largely on the computer's abilities (1) to construct systematically the addresses of the storage locations whose contents are to be processed, (2) to select the content of a storage location whose address has been constructed by the program, (3) to test the content of a storage location in various ways (for example, to see if it is the same as some reference code) and then to proceed with one of several alternative courses of further action that is selected by the outcome of this test, and (4) to record in a portion of the store the content of another storage location or a code that is associated with it in a reference table. A succession of codes that is constructed internally by these processes from a string of input codes, and recorded in another portion of the store, then can be "written" by further computer operations as output on another magnetic tape. This sort of process is used, for example, to form ρ-coded material from the Flexowriter input.

String transforming programs that produce a string of codes to be recorded as output by operating systematically on a string of codes that have been read as input are used in a variety of computer applications. The variants of the PC6 system that were used to process TYPRINT, TABPRINT, and other forms of input, come into this category, considered in their entireties. So do the preprocessors in these variant systems and the ρ-ω translator that was common to them all.

String transforming programs make extensive use of processes in which characters are examined, one by one, to determine where particular characters or sequences of characters occur. Such processes are called "scanning processes," though the word "scanning" does not imply any optical or visual process in this context but merely a programmed comparison of digital codes that have been stored in the computer.

7.3 Compilers

It is relatively difficult for programmers to construct the explicit sequences of machine language instructions that are needed for computer processes of any complexity. Programs are therefore written as sequences of "statements" in an "assembly" or a "compiler" language that is much easier for programmers to use. These statements are made up of letters, digits, and a few punctuation marks and special symbols. A program that is written in an assembly or a compiler language is called a "source program." A programmer usually writes the successive statements of a source program on the successive lines of a sheet of paper and then types the program on a keyboard perforator that punches paper tape or Hollerith cards. The tape or cards are then provided as input to the computer which translates the source program into the corresponding machine language program, by the use of a special program that is called an assembler or a compiler.

The PC6 programs were written almost entirely in the FORTRAN compiler language. This language was designed several years ago by a team of personnel from IBM and other organizations for use in arithmetic work. It was widely held for some time that FORTRAN would not be convenient for nonnumeric work such as the processes of string transformation. More recently, FORTRAN has been used for this latter type or work in many laboratories with considerable success. A statement in an assembler language is usually translated into a single machine language instruction by the assembler program, whereas a statement in a compiler language is translated into a sequence of several machine language instructions by the compiler program. A programmer who writes in an assembler language can control the computer's operations in much greater detail than a programmer who writes in a compiler language. Some programmers prefer to use compiler languages wherever possible while other programmers prefer to use assembler languages.

The author believes that compiler languages should be used wherever possible, because of the relative ease of writing and correcting programs and because of the attendant intelligibility of the programs to personnel who need to extend and adapt them. The implications of both these matters, which are of considerable economic and practical importance, are discussed further in Section 10.5.

One further aspect of programming may be mentioned here. This concerns the use of subprograms. The work of writing a large computer program may be compared with that of designing and manufacturing a complicated physical structure. In the latter activity, it is obviously expedient to construct an elaborate system from parts that can be manufactured separately. Each of these parts must conform to specifications that enable it to fit into the over-all structure, but which allow its further details to be changed if necessary. A complicated program deals in general with a complicated process that consists of several subprocesses. It is possible, then, to write a complicated program in separate parts. One part is called the main program, and has over-all control of the computers operations. The other parts are called subprograms and deal individually with the separate subprocesses which must be performed. The main program and the subprograms transfer control from one to another whenever necessary. The subdivision of a complicated process into the subprocesses for which separate subprograms should be written, and the corresponding specification of the subprogram characteristics are matters for the judgment of the programmer and often require much more experience than is needed to write the individual subprograms once their specifications have been provided in this way.

There are several advantages to the fragmentation of a program into subprograms in the manner just described. To run a program that is written in this way, each subprogram in the source language that the programmer uses is compiled or assembled by the computer into its translation in machine language, and this is recorded on punched cards that are called the "object deck" of the subprogram. To run the program, the object decks of the concomitant subprograms are loaded into the computer followed by the data. The object deck of a subprogram can be replaced by an alternative version without altering the object decks of any of the other subprograms. It follows that if a small portion of a program must be changed, for example to correct an error, then only the subprogram that contains the changed portion has to be reassembled or recompiled. This takes much less time than the recompilation of the entire program.

Subprograms provide a further benefit in that they facilitate the development of elaborate programs in stages. It may be possible to ignore some of the subprocesses in a complex process when a prototype program is written. Statements that call subprograms to perform these subprocesses can then be included at the points where these subprocesses will subsequently be brought into operation. To test the over-all program structure, however, use can be made, instead of the subprograms for the subprocesses which are being ignored, of dummy subprograms that simply return control to the calling subprogram as soon as they are entered. Then when the other portions of the program have been made operational, active versions of the subprograms can be written to deal with the subprocesses previously ignored. Subprograms also allow a modular approach to many problems; that is, a collection of subprograms may constitute a set of "building blocks" that can be used in different combinations for different jobs. The way in which the PC6 system of typesetting programs was organized as a collection of subprograms was one of its main characteristics.

7.4 Preprocessors

The fragmentation of a large program into subprograms is illustrated by the subdivision of each variant of the PC6 system into a preprocessor and the ρ-ω translator. Each of these preprocessors and the translator consisted of a major subprogram and numerous subsidiary subprograms that were brought into operation by one another or by the major subprogram, whenever necessary.

The TYPRINT preprocessor consisted of three groups of subprograms. The first group read the input into the computer from the input magnetic tape, rewrote the text material on a second magnetic tape in a form that was a little more convenient for further processing, and rewrote any editing instructions that followed the text onto a third magnetic tape. The second group of subprograms in the TYPRINT preprocessor performed the editing, if there were any editing instructions to be executed, and wrote the edited text onto magnetic tape in the form that had been used for the unedited text. The third group of subprograms translated the input (which had been edited if necessary) into the ρ-code.

The editing portion of the preprocessor contained separate subprograms to decode the editing commands, to locate the portions of the text that had to be changed, and to make the actual changes. An alteration in the conventions for writing the editing commands thus could have been accommodated by a revision of the subprograms that decoded these, without altering the other subprograms that were involved in editing if the types of action that the editing commands requested were not changed. A complete revision of the editing procedures could have been accommodated by a revision of the entire set of editing subprograms, but without altering the rest of the preprocessor.

The portion of the preprocessor that performed the translation from Flexowriter code to ρ-code consisted of several groups of subprograms. These were for (1) dealing with end-of-line spaces, blank lines, and tabulate codes (see Section 3.4); (2) delimiting control codes by reference to square brackets and commas; (3) translating mnemonic control codes that had been delimited in this way into their ρ-coded equivalents by a table look-up operation; and (4) translating text character codes into their ρ-coded equivalents by another table look-up. In this way, the conventions for end-of-line spaces, blank lines, tabulated indentions, and delimiting control codes, the format of the control codes, and the individual character and nonprinting codes of the keyboard perforator that was used to type input could all have been changed independently, by the revision of just one relevant subprogram. Further small changes could have been made to consolidate certain combinations of characters, such as ff, into the sequence numbers of appropriate ligatures on the disc.

The BCDPRINT preprocessor had to deal with very simple input operations and did not do any editing, so it consisted essentially of a group of subprograms that paralleled the third group of subprograms of the TYPRINT preprocessor which has just been mentioned.

The TABPRINT preprocessor consisted of two groups of subprograms. The first group read the specification cards and stored the typographic control information from these cards in coded form. This involved some very simple scanning operations to delimit control codes, page dimensions, headings, and so forth. The second group of subprograms read one by one the cards (or tape records) containing table entries, formed the ρ-codes for successive lines of the typeset table, and wrote these on magnetic tape. To do this, the portions of each input record that fell between columns which were stated on the specification cards were translated character by character into ρ-code. An appropriate string of typographic control codes, which was derived from the specification cards, was recorded in the ρ-string before the codes for each table entry, to provide the necessary horizontal space. The ρ-codes for page and column headings were recorded in the output at intervals that the computer determined by counting cards (or tape records) and referring to the specified page dimensions. Since the table cards did not contain control codes, the processing of these cards was much simpler than the processing of text matter by BCDPRINT and TYPRINT, which involved scanning operations to delimit and identify control codes.

The *Union List* preprocessor was specially designed to deal with the conventions which were adopted in the typing of the bibliographic data for this publication. Scanning operations were performed to recognize the combinations of carriage returns and to recognize individual characters such as parentheses, which were typographically significant. These operations were made more and more complicated as exceptions were found to the structure of the input items that was originally envisaged. As a result, the program became rather involved. The preprocessor translated each input character into ρ-code and inserted the ρ-code representation of the appropriate typographic control codes at the points which were determined by the scanning operations. The preprocessor also deleted the subject classification numbers that were included in the input and provided consecutive numbering of titles. The subroutine that translated input characters into ρ-code dealt with letters and digits in the same way as the corresponding subroutine in the TYPRINT preprocessor. It also translated the Flexowriter codes that corresponded to certain mathematical symbols on the keyboard into the Photon disc level and sequence number codes that were needed to set diacritical marks. Provision was also made in the *Union List* preprocessor for some editing operations to facilitate the correction of errors.

7.5 Mechanized Syntactic Analysis

The C.C.L. typesetting programs used preprocessors that performed scanning operations to determine the points at which typographic control codes should take effect. Further scanning operations delimited and identified these codes when they were included explicity as in TYPRINT source material. Direct FORTRAN programming can be used without difficulty for this sort of work, when dealing with input conventions such as those of TYPRINT and with input that contains simple combinations of carriage returns and tabulate codes to indicate cyclic changes of font. Much more elaborate scanning problems can arise, however, in the operation of some typesetting preprocessors. At times it is necessary to exploit the fact that particular combinations of characters start or end portions of the input that are to be distinguished typographically. These combinations of characters may be quite complicated and subject to numerous variations. To deal with such situations conveniently, use can be made of the rather powerful computing technique of mechanized syntactic analysis. The nature of this technique is illustrated in this section by reference to the preprocessor that was written to set the *Current Serials and Journals in the M.I.T. Libraries*, which is described in Section 6.6. In this job, the computer read a deck of cards in which the catalog was punched in Hollerith code. This code does not distinguish upper- and lower-case letters. The computer determined which letters were to be capitalized in the typeset version of the catalog by reference to some rules that are stated in Section 6.6. It was expedient to apply these rules by means of the "SHADOW" syntactic analysis program that was developed by the author some years ago for work of a rather different nature.[19] The "syntax" of the catalog entries is simply a description of the structure of these entries expressed in a particular stylization for reasons that are explained at the end of Section 7.6. In the discussion that follows, the name "C-string" is used for a portion of a journal title which consists of just a single letter which must be capitalized in the typeset product or which consists of such a letter followed by one or more characters that do not include any further letters to be capitalized. The name "N-word" is used for a word which does not contain any letters that should be capitalized.

The syntax of the journal titles that are punched in the Hollerith card deck of the *Current Serials and Journals* catalog can be defined as follows:

A title consists of a C-word then possibly a letter-free string, then a sequence of zero or more strings each of which consists of a word then possibly a letter-free string.

A C-word consists of a sequence of one or more C-strings, then a separator.

A C-string consists of a letter then zero or more alphameric characters, then possibly either a period or a hyphen.

An alphameric character is either a letter or a digit.

A separator is either a blank or a left parenthesis or a right parenthesis.

A letter-free string is a sequence of one or more characters none of which is a letter.

A word is either an N-word or a C-word.

An N-word is either "throughout" or "following" or "divided" or "parts" or "entries" or "issued" or "before" or "under" or "bound" or "thru" or "only" or "also" or "with" or "into" or "och" or "aus" or "van" or "etc." or "per" or "die" or "des" or "und"

or "der" or "zur" or "fur" or "and" or "for" or "the" or "of" or "in" or "de" or "do" or "es" or "et" or "le" or "la" or "on" or "di" or "si" or "po" or "ed" or "ee" or "du" or "by" or "to" or "i" or "l" or "u" or "y" or "e" or "s" or "d" or "a", then a separator.
A period is ".". A hyphen is "-". A left parenthesis is "(".
A right parenthesis is ")". A blank is " ".

This defines the syntax of a title in terms basically of letters, digits, characters (i.e., all letters, digits, and symbols, and blanks, in the Hollerith code), and the sequences of characters that are quoted explicitly in the last six sentences of the definition table. The SHADOW program analyzes a title by reference to this syntax in the following way. It begins by examining the sentence that defines a title. This refers to a C-word. The program therefore transfers attention to the sentence that defines a C-word and thence to the sentence that defines a C-string. Then, in accordance with this latter sentence, it tests if the first character in the input is a letter. Unless the input is in error, this test succeeds. The program then tests if the next character is an alphameric character, that is a letter or a digit. If the character is not alphameric, the program continues to the next portion of the definition. If the character is alphameric, the program tests successive characters in the input until one is found that is not alphameric. When the program has found the first character that is not alphameric, it tests if this is a hyphen or period. If the character is a hyphen or a period, it is included in the portion of the input that is identified as a C-string; otherwise it is not included.

The program then returns attention to the sentence that defines a C-word. This immediately transfers attention again to the sentence that defines a C-string, and the program tests if another C-string follows that which has just been delimited. This is the case, for example, in an abbreviation such as A.E.G. The program cycles back and forth between the sentences that define a C-word and a C-string until a break occurs in the sequence of contiguous C-strings. The program then tests if a separator occurs at the position in the text which has been reached in the scanning process, in accordance with the sentence that defines a C-word. Unless the input is in error, a separator does occur at this point and is identified, and attention then returns to the sentence which defines a title.

The program then tests for a letter-free string following the separator that follows the first C-word. If the title consists of just a single C-word, the scan delimits the trailing blanks in the title field as a letter-free string and then terminates. If the title consists of a single C-word followed in the rest of the title field by a mixture of blanks and other characters that include no letters, then these characters are identified collectively as a letter-free string, and the scan is ended.

When a letter occurs at some position in the title beyond the end of the first C-word, the program delimits a C-word, then a letter-free string, and then tries to delimit another word. To do so, it tests if an N-word can be delimited. Each of the words that is quoted in the definition of an N-word is considered in turn. If one of the quoted words matches the leading characters in the portion of the input that remains to be analyzed, the program then tests if a separator (i.e. blank or bracket) follows. If this test is successful, the string of characters is accepted as an N-word. If the test is unsuccessful, however, the program attempts to delimit a C-word from the beginning of this string. Another C-word in fact would be delimited if the input was valid. The program would then attempt to delimit another letter-free string. If only blanks and other characters that included no letters were left in the title field, the scan would delimit these collectively as a letter-free string and terminate. The further processes that would occur if another letter were encountered would just be a repetition of those which have been described already for the delimitation of a word.

It should be noted that a word, such as "THE", which is treated as an N-word when it occurs internally, is treated automatically by the program as a C-word when it begins a title, as a consequence of the sentence that defines a title in the definition table. For later sequences of letters in a title, the requirement that a separator follows a quoted string to form an N-word ensures that the letters which can be one letter connective words and the sequences of letters such as "LA" that form connective words are recognized as abbreviations when followed by a period. Moreover, "ETC." is recognized as an N-word by the program even though it ends with a period, because the latter is identified within the test for a N-word, which simply compares successive characters in the quoted strings with successive characters in the text.

The results of the syntactic analysis are recorded internally in a table that lists the names of the syntactic units which are delimited during the analysis. Each name is accompanied by the enumeration of the first and last characters of the portion of the input to which the name applies. Each letter to be capitalized is the first character of a C-string. The program which produces a representation of the title that includes case shift indicators then has merely to refer to the entries in this table for the beginnings of the C-strings to determine the positions of the characters that are to be capitalized.

The program that was used to set portions of the *Current Serials and Journals* catalog in the C.C.L. worked in effectively the manner that has been described, though the details have been simplified a little in the present account for ease of explanation. It is quite likely that the rules which have been incorporated in the definition table for a title would be found to be inadequate if the job were run on a continuing basis. The list of quotations in the sentence that defines an N-word might require expansion to include connectives in further languages. It can happen, too, that a sequence of letters which forms a connective word that should not begin with a capital letter in one language also forms a word that is of greater importance and which should begin with a capital letter in another language. For example, *fur* is a connective in German and a noun in English. This circumstance can be accommodated by extending the definition table to distinguish the uses of the sequence by reference to the context in which it occurs.

The point to be stressed is that the extension of the definition table and the development of corresponding definition tables for other problems require no knowledge of computer programming. All that is needed is an ability to express the scanning rules in words, in the style that is illustrated by the definition tables that are given in this chapter and in other accounts of the SHADOW syntactic analysis program.[19] This is in contrast to the situation that was described in Section 7.4 for the TYPRINT and TABPRINT preprocessors, in which the changes of scanning procedure that are needed to deal with changed input conventions do require the revision of computer programs by personnel who can program.

7.6 Syntactic Analysis and Stylized Instructions

An experimental editing system that was called the ES1 system provides another example of the use of syntax definition and mechanized syntactic analysis.[9] The ES1 system was started in the C.C.L. before the typesetting project began. It was of no practical value as an editing program since it could deal only with texts of rather limited length. The way in which editing instructions were expressed for input to the program, however, is of some interest. A stylized form of English language sentence was used to describe the editing changes which were to be applied to a text.

The style of the editing instructions used with the ES1 system is illustrated by the following examples, which refer to a portion of "Alice in Wonderland" that was used to test the program:

Delete and close the first word in the third line of the second paragraph.
Delete the last sentence in the third paragraph.
Delete and close from the fifth paragraph through the second occurrence of a paragraph beginning with the name "Alice".
Delete and close the expression "and then unrolled the parchment scroll".
Delete "she made some tarts. All on a".
Insert "and precisely" before the fourth line in the eighth paragraph.
Run on the first paragraph.
Start a new line with the last sentence in the seventeenth paragraph.
Start a new paragraph with the last two words in the fourth line of the seventh paragraph.
Start a new page with the paragraph ending with the expression "the king".
Indent the second sentence in the eighth paragraph.
Start a new line with the fourth word in the twentieth paragraph.
Exchange the last four words in the second line of the sixth paragraph with the first two words in the paragraph ending in the word "talking".
Exchange the eighth occurrence of the word "King" with the word "Hatter" in the twenty-first paragraph.
Exchange the word "Hatter" with the word "king" in the twenty-third paragraph.
Exchange the tenth paragraph through the paragraph ending in "away" with the thirteenth paragraph.
Exchange "'Not yet, not yet!'" with "the Rabbit hastily interrupted.".
Replace the word "the" in the thirteenth paragraph with "The".
Replace "!" in the thirteenth paragraph with ".".
Replace "Not" in the thirteenth paragraph with "not".

It can be seen that each of these instructions is a simple sentence which consists of an imperative verb, followed by an object which contains references to one or more portions of the text, and perhaps a quotation for inclusion in the output.

The further details concerning the ways in which the editing instructions were constructed could be gathered from inspection of the larger collection of examples that was provided in the account of the

ES1 system. The term "stylized instruction language" is used for a stylized subset of a natural language, that is used to express instructions concerning some computer application in a form which can be understood by personnel who have not had to learn any conventions concerning the preparation of input, and which can be interpreted by a computer program. In general, information that is expressed in a stylized instruction language first must be reduced to a codified form by the computer before the action that it requests is performed. The extraction of the essential items of information from a stylized instruction can involve elaborate scanning operations that are facilitated by the use of a syntactic analysis program provided with the syntax of the stylized instruction language for reference. Moreover, the use of a syntax to define the structure of the input instructions makes it relatively easy to expand and vary the details of the stylized instruction language by changing the syntax without altering the program. The syntax for the ES1 instruction language is reproduced as Table 7.1. It provides a good example of the way in which the syntax for a stylized instruction language is constructed.

7.7 The ρ-ω Translator

Many of the details of the ρ-ω translator were specifically designed for the Photon 560 machine and for interim measures that were adopted in the course of what was essentially an exploratory activity. A few comments may be made, however, about the general organization of the translator that may be of use in future programs. These comments also provide a further illustration of the fragmentation and open-endedness that are so important in program evolution. The translator was organized in a "layered" structure which is depicted diagrammatically in Figure 7.1. Each rectangle represents a subprogram, and control was passed upward and downward as necessary between subprograms whose rectangles abut each other vertically in this diagram. This sort of organization makes it relatively easy to change, say, an end-of-page procedure to modify the page numbering conventions without affecting the rest of the program. The translator operated on a block of ρ-codes in the main store of the computer that could be only a fraction of the length of the entire body of ρ-codes that was to be processed. The program therefore was constructed so that the translation could be interrupted during the operation of any subprogram depicted in the central part of Figure 7.1, and control passed upward through successive layers of the translator to the subprogram depicted at the top. This replenished

Table 7.1. *Verbalized definition table for ES1.*

An editing instruction is either a type 1 editing instruction or a type 2 editing instruction or a type 3 editing instruction or a type 4 editing instruction or a type 5 editing instruction or a type 6 editing instruction or a type 7 editing instruction or a type 8 editing instruction or a type 9 editing instruction or a type 10 editing instruction or a type 11 editing instruction or a type 12 editing instruction.

A type 1 editing instruction consists of "Delete" then a gap then "and" then a gap then "close" then a gap then a mixed reference then a period.

A type 2 editing instruction consists of "Delete" then a gap then a mixed reference then a period.

A type 3 editing instruction consists of "Insert" then an arbitrary phrase ended by a quotation, then an arbitrary phrase ended by "before", then a gap then a reference then a period.

A type 4 editing instruction consists of "Insert" then an arbitrary phrase ended by a quotation, then an arbitrary phrase ended by "after", then a gap then a reference then a period.

A type 5 editing instruction consists of "Replace" then a reference then "with" then a quotation then a period.

A type 6 editing instruction consists of "Run" then a gap then "on" then a reference then a period.

A type 7 editing instruction consists of "Start" then a gap then "a" then a gap then "new" then a gap then "line" then a gap then "with" then a gap then a reference then a period.

A type 8 editing instruction consists of "Indent" then a gap then a mixed reference then a period.

A type 9 editing instruction consists of "Exchange" then a gap then a mixed reference then "with" then a mixed reference then a period.

A type 10 editing instruction consists of "Set" then a gap then "margin" then an arbitrary phrase ended by a numeric, then an arbitrary phrase ended by a numeric, then a period.

A type 11 editing instruction consists of "Start" then a gap then "a" then a gap then "new" then a gap then "paragraph" then a gap then "with" then a reference then a period.

A type 12 editing instruction consists of "Start" then a gap then "a" then a gap then "new" then a gap then "page" then a gap then "with" then a reference then a period.

A mixed reference consists of a reference then possibly “through” then an arbitrary phrase ended by a reference.

A reference is an arbitrary number of phrases each of which is an arbitrary phrase ended by a reference unit.

A reference unit is either a quotation or either “last” or “first”, then a gap then a cardinal number or “last” or an ordinal adjective, then a gap then either “paragraph” or “sentence” or “line” or “word”, or “the” then either “paragraph” or “sentence” or “line” or an ordinal adjective then an occurrence, or a start or a finish.

A quotation consists of “ “ ” then an arbitrary phrased ended by “ ” ”.

A gap is a sequence of one or more symbols each of which is either a tab or a space or a carriage return.

An occurrence consists of either “appearance” or “occurrence”, then a gap then “of” then a gap or “time” then a gap then either “it” or “they”, then a gap then either “occur” or “appear”.

A start consists of either a pronoun then a gap then either “starts” or “begins”, or a starting participle, then a gap.

A finish consists of either a pronoun then a gap then either “finishes” or “ends”, or an ending participle, then a gap.

A numeric is a sequence of one or more symbols each of which is a digit.

A pronoun is either “that” or “which”.

A starting participle is either “starting” or “beginning”.

An ending participle is either “finishing” or “ending”.

A digit is either “0” or “1” or “2” or “3” or “4” or “5” or “6” or “7” or “8” or “9”.

A cardinal adjective is either a cardinal digit or “ten” or “eleven” or “twelve” or “thirteen” or “fourteen” or “fifteen” or “sixteen” or “seventeen” or “eighteen” or “nineteen” or “twenty” or “thirty”.

A cardinal digit is either “one” or “two” or “three” or “four” or “five” or “six” or “seven” or “eight” or “nine”.

An ordinal adjective is either an ordinal digit or either “twenty” or “thirty”, then an ordinal digit or “eleventh” or “twelfth” or “thirteenth” or “fourteenth” or “fifteenth” or “sixteenth” or “seventeenth” or “eighteenth” or “nineteenth” or “twentieth” or “thirtieth”.

An ordinal digit is either “first” or “second” or “third” or “fourth” or “fifth” or “sixth” or “seventh” or “eighth” or “ninth”.

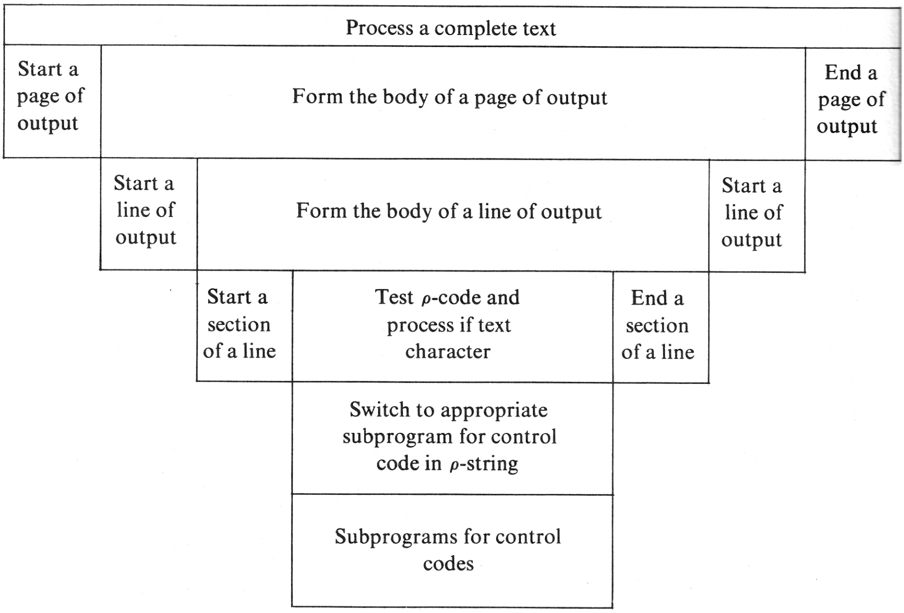

Process a complete text
Start a page of output
Form the body of a page of output
End a page of output
Start a line of output
Form the body of a line of output
Start a line of output
Start a section of a line
Test ρ-code and process if text character
End a section of a line
Switch to appropriate subprogram for control code in ρ-string
Subprograms for control codes

Figure 7.1. *Diagrammatic representation of structure of ρ-ω translator.*

the supply of ρ-codes to be translated, by reading more from the magnetic tape on which they were stored, and returned control downward through the appropriate number of layers to the subprogram whose operation had been interrupted, and which then continued the translation where it had left off.

The PC6 programs used about 50 reference items or parameters to keep track of the typographic requirements and progress of the job being run. Some of these were modified very often; for example, the parameter that represented the space available for further characters on an output line was modified whenever a text character was processed. Other parameters were modified less often; for example, the parameter that represented the point size was modified only when the ρ-code equivalent of an LS code was encountered. Some of the parameters were "pointers"; that is, they enumerated a character that was of interest for some reason in the input or the output string, for example, as the ending of the last full text word to be processed, to which the program would have to redirect action if the word that followed it caused overset. Several subprograms were concerned largely or in part with retroactive and backtracking procedures. Retroactive procedures, for example, record information that is generated at one juncture, in the context of material generated previously, as must be done when interword spaces for an output line are finalized only when the last word to be set in that line has been processed. Backtracking procedures are invoked when the tentative processing of some material produces unsatisfactory consequences (such as overset) and material has to be processed afresh and in an alternative manner. Retroactive and backtracking procedures tend to be much simpler and faster when they refer or revert to material that is still in the main store of the computer. A computer with a very small main store can retain material for a line of output. Most modern stored program computers have the capacity to retain a page of material, as is necessary for page justification. Backtracking procedures that affect still larger bodies of material may arise, for example, in connection with the positioning of runarounds for illustrations.

The ρ-ω translator was organized so that each ρ-code was tested by the subprogram which is designated "test ρ-code and process if text character" in Figure 7.1, and which is almost at the base of the structure that this diagram represents. When a control code was encountered, control was passed downward to a "switching" subprogram, which in turn brought into operation a further subprogram that was appropriate to the control code's identity. This latter subprogram reset the relevant parameters in the necessary manner, and then passed control upward to the level that was appropriate to the action which had to follow recognition of the code. This program structure was planned as a method for leaving the set of control codes open ended. Several further control codes could have been accommodated by addition of subprograms to reset parameters in accordance with their requirements and by a trivial expansion of the switching subprogram. In programming terminology, the control codes constituted a set of special-purpose operation codes that were used interpretively.

7.8 Further Typographic Functions

Emphasis has been placed, in several sections of this monograph, on the distinction between the typographic functions that a typesetting program can exercise and the style of coding in which a user expresses typographic requirements. It seemed helpful to introduce the typographic functions that the PC6 programs perform by reference to the TYPRINT control codes which are used to bring these functions into operation. This is done in Chapter 3, which develops the idea of interspersed control codes. The possibility of changing the form of the control codes without altering their functions is explained in Chapter 4. The possibility of omitting control codes from the input and using suitable preprocessors to introduce these mechanically is exemplified by the treatment of TABPRINT material that is described in Chapter 5 and by some of the other applications of the PC6 programs that are described in Chapter 6. These are all limited, however, to the typographic functions that characterize the PC6 programs, regardless of the way that the use of these functions is requested, either explicitly or implicitly. Some suggestions concerning further typographic functions that would be useful are presented in the paragraphs that follow. For simplicity, the functions are described by reference to control codes in the style of TYPRINT that might be used to bring them into operation. The TYPRINT style is retained in this discussion, however, only for ease of presentation. The possible utility of the functions should not be measured by the convenience or otherwise of the TYPRINT style when preparing input, since different forms of representation could be provided with relative ease once the subroutines that performed the requisite typographic functions had been written.

1. *Paragraph indention.* A control code of the form [PI*m*], where *m* stands for an integer, could be used to specify that *m* relative units of indention should be provided before the first word of each paragraph in a text, until a further PI code was encountered. The PC6 programs provided 18 relative units of indention automatically.

2. *Interword space.* A control code of the form [IW*m*], where *m* stands for an integer, could be used to specify that *m* relative units of space should be provided tentatively for each input space, subject to alteration in the justification process. The PC6 programs provided 6 relative units of space for this purpose.

3. *Letter space mode.* A control code of the form [SL] could be used to request letter spacing to justify lines that otherwise would contain interword spaces in excess of the tolerance defined by the HI code. When the PC6 programs were used, the tolerance was always made so great that it was not exceeded in practice. If letter spacing were provided by the program, the HI code could be used to limit interword spaces to a typographically acceptable tolerance. The further expansion of lines of text would be provided, when necessary, by increasing the escapement to give additional space between letters. Another code, say [WS], would be needed to suppress letter spacing if it were undesirable for particular jobs.

4. *Set.* A code of the form [STx], where x stands for a decimal number, could be used to specify the use of a set size that is x times the point size in current use. Alternatively, a code of the form [SEy] could be used to specify a set size of y points. The significance of set size is explained in section 2.11.

5. *Running heads and footnotes.* Codes [SH] and [EH] could be used to delimit the beginning and end of a running head, respectively. It would be necessary to allow control codes for type size, disc level, and so forth, to follow the SH code and to occur within the text matter of the running head, and for the EH code to restore the status of these functions to that which was operative before the SH code was encountered. A code of, say, the form [RHm], where m stands for an integer, would be needed to specify the distance in points between the running head and the first line of text. The lateral position of the running head could be specified by an FL, FC, or FR code after the SH code. Some convention would be needed concerning the action that the program should take for running heads of excessive length. An error condition could be signaled, or the heading set on two or more lines. Provision would also be needed to combine running heads with page number when the latter was requested at the top of the page in a way that did not conflict with the presence of a heading. The inclusion of running heads is affected also by the expand code EX described below. Codes of the form [SF] and [EF] could be used to denote the beginning and end of material to be set as a footnote on the page of typeset output which contains the material that precedes the [SF] code.

6. *Expand.* A code, say [EX], could be used to request the expansion of interword spaces in a line of text to make it fill the specified measure even though overset had not been caused by a subsequent word. The EX code would be included in the input, immediately after the material to be expanded. For example, input that was typed

[LN360,LS6,NL] Example of underset expansion. [EX,NL]

would produce the sentence "Example of underset expansion", set in 6-point type, and with uniform interword spaces that made the initial "E" and terminal period 360 points apart. If the EX code followed text matter such as a paragraph which was set on several lines, all but the last of these would fill the requisite measure as a result of the usual justification procedure (if the justification mode was operative) and the last line would fill the measure as a result of the EX code.

7. *Leaders.* A code, say [LR], could be used to request the in-

clusion of enough leaders (dots) between the portions of text that it separated, to fill out the measure of the line or section being set. The inclusion of the [LR] code would be meaningful if a further code, such as [NL], [NP], or [XS], followed soon after text matter on the right of the [LR] code. Thus the input

[LN360,LS6,NL] England [LR] London [NL]
U.S.A. [LR] New York [NL]

would produce output set in 6-point, with the words England and U.S.A. starting from the left margin on two successive lines, the words London and New York on these two lines flushed right to a margin that gives a 360-point measure, and leaders (dots) filling out the space between the country and town names on the two lines. It would be necessary to establish some further conventions to ensure that leaders were aligned vertically and to allow variety in their spacing. The code could be expanded, say to the form [LR*n*], where *n* took values 1, 2, 3, etc., for single dots, double dots .. .. .., triple dots, and so forth, that were spaced by varying amounts. To ensure vertical alignment, the first leader in any line could be placed so that its distance from the left margin in current use was an integer multiple of the relative width assigned to the leader.

8. *Restart page number.* A code of, say, the form [RP*n*], where *n* stands for an integer, could be used to restart page numbering by assigning *n* to be the page number of the next new page to be started after the code was encountered.

9. *Relative margin setting.* A code of, say, the form [LC*n*], where *n* stands for an integer, could be used to set the left margin of section *n* through the position in the typeset output at which a character would have been set if it had been encountered instead of the [LC*n*] code. More precisely, [LC*n*] would indent the left margin of section *n* from the left edge of the film by the sum of all the escapements that were recorded since the last end of output line. Another code, say of the form [RC*n*], where *n* stands for an integer, could be used to set the right margin of section *n* at the position where the left margin would be set by the [LC*n*] code in the same input context. In connection with the codes that are described in this paragraph, it would be essential to remove the restriction of the PC6 programs that sections may not overlap.

10. *Nonset mode.* The relative margin setting codes [LC*n*] and [RC*n*] of the preceding paragraph would be particularly powerful if used in conjunction with a nonset mode code, say [NS], and a cancel nonset mode code, say [CN]. These codes would precede

and follow, respectively, a portion of text which was typed in the input for the purpose of determining margin settings in the output, but which was not to appear in the output in the same context. As an example of the use of these codes, margin settings for a table of *m* columns could be established by typing [NL, NS, LC1], then the longest entry in the first column, then [RC1,LC2], then the longest entry in the second column, then [RC2,LC3], and so on, with [RC*m*,CN,NL] after the longest entry in the last column. In addition to the [NS] code, a further code would be useful, say [BS], which differed from the [NS] code in that it spaced backward. The [BS] code would be used to set a margin by spacing backward from some other reference position to leave room for the material of the width that was typed between the [BS] and [CN] codes.

11. *Blocks.* Some codes, say of the forms [(UB$n=x$, LB$n=y$)] and [XB*n*], could be used to control the setting of output in vertical blocks in a manner analogous to the use of the LM, RM, and XS codes to control the lateral arrangement of material. The word "block" is used here for a portion of a page that is bounded by upper and lower margins whose positions have been specified, for example by UB and LB codes. The blocks would be numbered, and the codes of the form [(UB$n=x$, LB$n=y$)] would define the upper and lower margins of the block. The code [XB*n*] would request that the material which followed it in the input should be set in block *n* of the output. This would be of use, for example, when one portion of a typeset page had to start from a particular horizontal line regardless of how far above this line the previous material ended. Further codes and corresponding programming to provide page justification, vertical centering, lowering, and expansion of material with respect to horizontal margins would be of use in particular circumstances.

12. *Runarounds.* Several codes would be needed to specify the details of runarounds. Codes of, say, the form [HR*x*] and [WR*y*], where *x* and *y* stand for numbers, could be used to specify the height and width of a runaround, respectively. The placing of the runaround at the left and right of the page could be specified by two further codes, say [LR] and [RR], respectively. The HR, WR, and LR or RR codes could be written in sequence together with a further code that specified the direction in which the runaround should be moved if it could not be started at the point in the text at which the codes were typed. The mnemonics [MU] and [MD] could be used to show that the runaround should be moved up and down, respectively.

13. *Delay modifiers.* It would be of use to be able to request that

one or more typographic functions come into operation after a specified delay that was determined by the progress of the setting process. Thus a code of, say, the form [WL*n*], where *n* stands for an integer, could be used to delay for *n* lines of typeset output the effect of further codes which followed it in brackets. Thus [(WL2:LM24)] would indent the left margin by 24 points after a delay of two lines in the typeset product. Similar codes of, say, the forms WP*n* and WG*n* could be used to delay the effect of a string of typographic functions by *n* points and by *n* pages respectively.

It may be mentioned that the PC2 typesetting programs, which were written in the C.C.L. before the PC6 system was developed, did contain some useful features, such as codes analogous to those discussed in paragraphs 9 and 10, which were omitted from the PC6 system.[20]

7.9 External Control of Cyclic Format

The section modifier cards that were included in TABPRINT specification decks were said to provide external control of cyclic format because the cards were used to define a cycle of format changes that repeated from line to line in the typeset product, and because the cards were all read by the computer before the material that was to be typeset was read. This style of format definition was used in TABPRINT input for convenience. It would have been possible in principle (though pointless) to use an ordinary tabulator to print the deck of cards in which a table was punched, to annotate the listing by hand with TYPRINT control codes, and to type the annotated listing on a Flexowriter, producing a punched paper tape that was used as TYPRINT input. All the control codes that were needed to effect the cyclic changes in format would have been interspersed in this TYPRINT input at the points where they were to take effect, and it would have been possible (though pointlessly time consuming) to set the table without invoking external format control. This possibility would have resulted from the fact that the positions in the text at which typographic changes had to take effect were known *before* the typesetting codes had been formed by the computer. There are further situations, however, in which cycles of typographic changes must be repeated in the typeset product, and the positions of these changes cannot be anticipated before the typesetting codes are formed. To deal with these situations, external format control is essential. A style that could be quite useful is described now. It has several similarities to the style of FORMAT statement which is used in the FORTRAN language[4] to provide, in effect, external control of material that is printed on conventional computer output devices.

Let X and Y stand for two sequences of TYPRINT style control codes that had to take effect in alternation for *n* and *m* lines, respectively. This could be denoted by a typographic control expression of the form (*n* lines (X), *m* lines (Y)). If the sequences of control codes that are denoted here by X and Y had to take effect in alternation for *n* and *m* points, respectively, this could be denoted by (*n* points (X), *m* points (Y)). In general, expressions of the forms *n* lines (X), *n* points (X), *n* pages (X) could be used to request that the TYPRINT style codes denoted by X remain in force for *n* lines, points, and pages, respectively. A sequence of two or more such expressions separated by commas, with a pair of brackets around the entire sequence, could request the repetition of a basic cycle that consisted of the execution of each of the expressions within the brackets in turn. Larger cycles could be defined by a sequence of two or more expressions with an integer, or an integer and the word "lines," "points," or "pages," if meaningful and appropriate, before each of these and a pair of brackets around the entire sequence. Cycles of arbitrary depth could be defined by extending the formation of these expressions to the appropriate depth of bracketing.

7.10 A Procedural Language

Although internal coding of the forms described in Chapter 3 and Section 7.8 and external coding of the forms described in Chapter 5 and Section 7.9 can be used to specify typographic functions for a considerable variety of typesetting jobs, there is a need, particularly in connection with complicated page composition, for a procedural language to express typographic make-up. Some of the elements of such a language are described in this section. They would provide a convenient way of handling certain of the TYPRINT style control codes that are mentioned in Section 7.8.

It would be necessary for the procedural language to allow code translation of several texts that had been read from an input medium such as Flexowriter tape, and to proceed piecewise, each translation being interrupted when a condition that was specified in the procedural language occurred. Input texts and translated texts would be identified by names that the programmer provided. Provision would be necessary for internally and externally coded input, and for the correct resumption of the processing of such material which had been interrupted. The instructions that called for the translation of material from the input code into the typesetting machine operating code would specify (1) the names that had been given to the input and output representations of the material in question, (2) pointers to the positions at which processing should continue in the arrays that stored these representations, and (3) the amount of input to be processed or of output to be set (e.g., residue of 1 line, *n* lines, *n* points horizontally, *n* points vertically, residue of 1 page, *n* pages, residue of input, etc.). It would be necessary also for the pointers to the positions in the input and output arrays at which processing was interrupted to be stored as the current values of variables to which the programmer assigned convenient names. Further instructions would be needed to count items of various types, such as words and lines, and to sum various properties, such as horizontal and vertical spacing increments between specified positions in the input and output strings. The facility to assign numerical values to variable names that appeared as arguments of typographic control codes would be needed, as well as all the arithmetic, input-output, and control facilities of an algebraic compiler language such as FORTRAN. It is possible that all the forms of instruction and other facilities mentioned in this section could be obtained by using the FORTRAN language as the procedural language, with CALL statements to appropriate subroutines acting as the typographic instructions.

8. Other Computer-Aided Typesetting Projects

8.1 Introduction

Interest in computer-aided typesetting is developing at a pace which would soon outdate any present survey. This chapter can do no more than mention a few selected projects to indicate the variety of possible applications of computers to typesetting and to related matters. In several of the projects which are mentioned here, a computer is used to perform tedious routine computational or clerical work, and a tape-operated typesetting machine is employed to set the results that otherwise would be printed by conventional computer output devices. In several of the other applications, a computer is used to allow keyboard operators to work faster because they no longer have to determine line and word breaks.

One of the earliest examples of computer-aided typesetting involved the use of a Linofilm photo unit to set the output of a Russian-to-English translation program. Similar typesetting equipment has also been used more recently to set some extensive numerical tables that were produced by a computer. These two applications are described in Sections 8.2 and 8.3, respectively. An academic study of mechanized hyphenation that has been in progress for several years is described in Section 8.4. Some of the text-setting programs that break lines and hyphenate mechanically for newspaper and book production, which have been developed by commercial organizations, are described in Section 8.5. A special purpose device that allows speedier keyboard operations is mentioned briefly in Section 8.6. A few comments concerning the *Index Medicus* project, which uses a powerful computer to perform clerical operations and a high-speed optical typesetting machine to set the results, are provided in Section 8.7. A projected high-speed electron-beam typesetting machine that will be operated by computer-generated magnetic tape is described in Section 8.8. Some further projects that make use of computer-aided typesetting are mentioned briefly in Section 8.9.

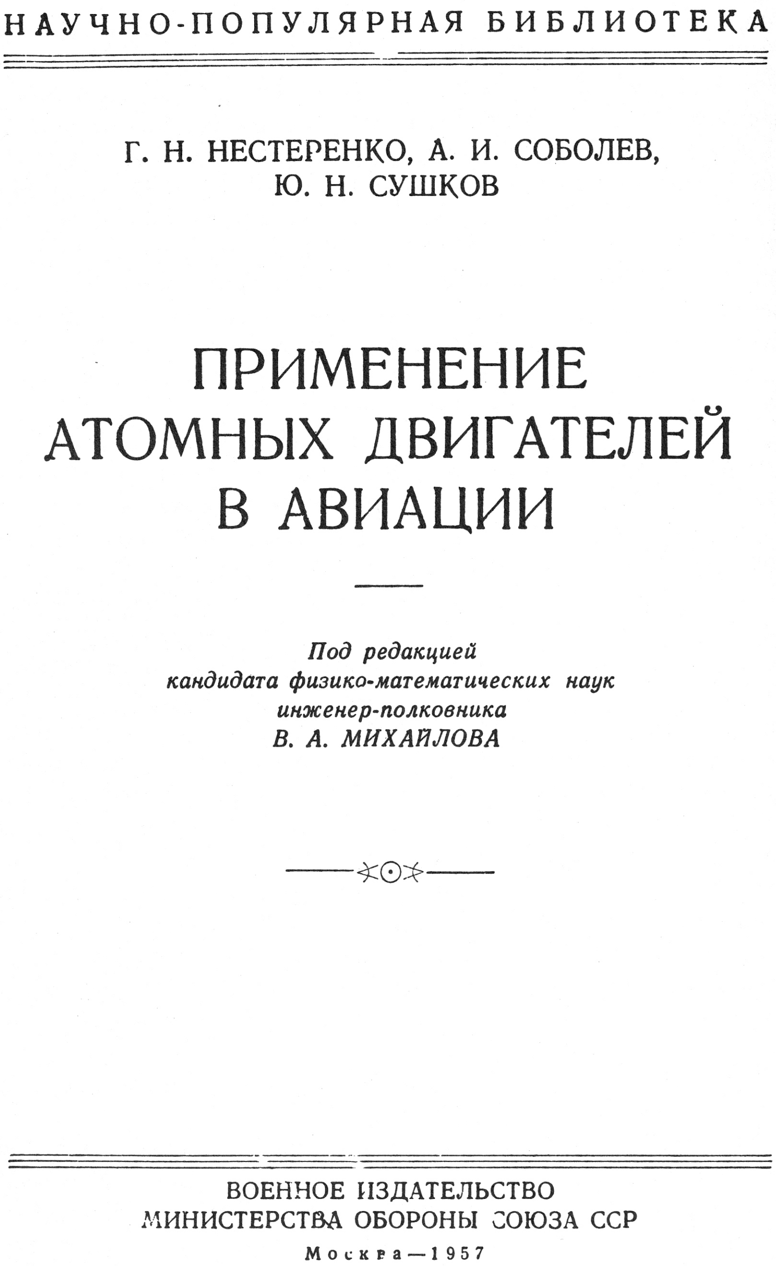
8.2 The Russian Translation Project

The possible use of digital computers to translate text material from one natural language into another has been investigated extensively in several countries during the last few years. One major project that was concerned with the translation of Russian technical material into English was supported at the Thomas J. Watson Research Center of IBM Corporation by the Information Processing Laboratory, Rome Air Development Center, United States Air Force Systems Command.[21] This project made use of an information processing machine designed specifically for text processing work, and organized around a device that is called the photoscopic store. Russian text material to be translated was keyboarded on a Flexowriter which typed Roman characters when in lower-case shift, and Cyrillic characters when in upper-case shift. The Flexowriter tape was used as input to the translation machine. This punched output into paper tape to operate another Flexowriter that typed the translation mechanically. This second Flexowriter typed both upper- and lower-case Roman letters. The output tape of the translation machine could also be used as input to an IBM 704 computer which then produced an operating tape for a Linofilm phototypesetting machine.[22]

The Linofilm machine projects images of illuminated characters one by one on a roll of continuous film, under the control of an operating tape, and in this general respect it is analogous to the Photon equipment which is described in Chapter 2. The Linofilm and Photon machines differ in the principles which are used to select and to magnify characters. Eighteen rectangular glass plates, which each carry the negatives of eighty-eight characters, are mounted in a "basket" in the Linofilm machine. Each plate (or "grid" as it is called) can be moved mechanically into an area which can be illuminated by a light beam. This beam is controlled to project the image of just one selected negative through a "zoom" lens on a roll of film in a magazine. The position of this lens along the optical axis of the system determines the magnification of the negative, and hence the point size of the type. Codes in the operating tape control the selection of (1) the grid that is in current use, (2) the negative that is illuminated, (3) the position of the zoom lens, (4) the lateral displacement of the projected character across the width of the film, and (5) the vertical movement of the film between lines. Paper tape that is punched by a special perforator is used to operate the Linofilm typesetting unit in noncomputer-aided applications. The typesetting unit in the Russian translation project was modified for operation by a computer-generated magnetic tape.

The Linofilm machine was used to set the translated material, so that it could be read with less fatigue than Flexowriter copy, and so that a wide range of mathematical symbols could be included in technical matter. The material which was translated was all taken from printed documents, and it was of some convenience to the users of the output that the general format of the source material was preserved in the final product. To achieve this objective, the Russian source documents were annotated with typographic control codes, using some simple conventions which are described below, and these codes were typed at appropriate points in the Russian text when the input to the translation machine was prepared. This machine translated the text into approximate English and also translated the typographic control codes into a second form of coding which was included in an output tape that operated the Roman character Flexowriter to type a copy for initial inspection of the text translation. The typographic control codes were represented in this tape by codes which did not activate Flexowriter type bars. The codes that represented special characters actually caused the Flexowriter to advance the carriage, leaving space for subsequent manual inscription. The output tape of the translation machine was used as input to a 704 computer, which produced the operating codes for the Linofilm machine. Editing corrections that were typed on a Flexowriter could be applied to the text translation at this stage by an editing section of the 704 program.

The conventions for preparing source material for the translation machine were very straightforward. Two consecutive ¢ symbols preceded each control code, and the code itself was delimited by its identification in a table look-up operation. Seven three-letter mnemonics were used to specify particular combinations of type size and style for main title (mtl), subtitle (stl), footnote (ftn), abstract (abs), author's name (aut), book title (btl), and text (txt). Two further



НАУЧНО-ПОПУЛЯРНАЯ БИБЛИОТЕКА

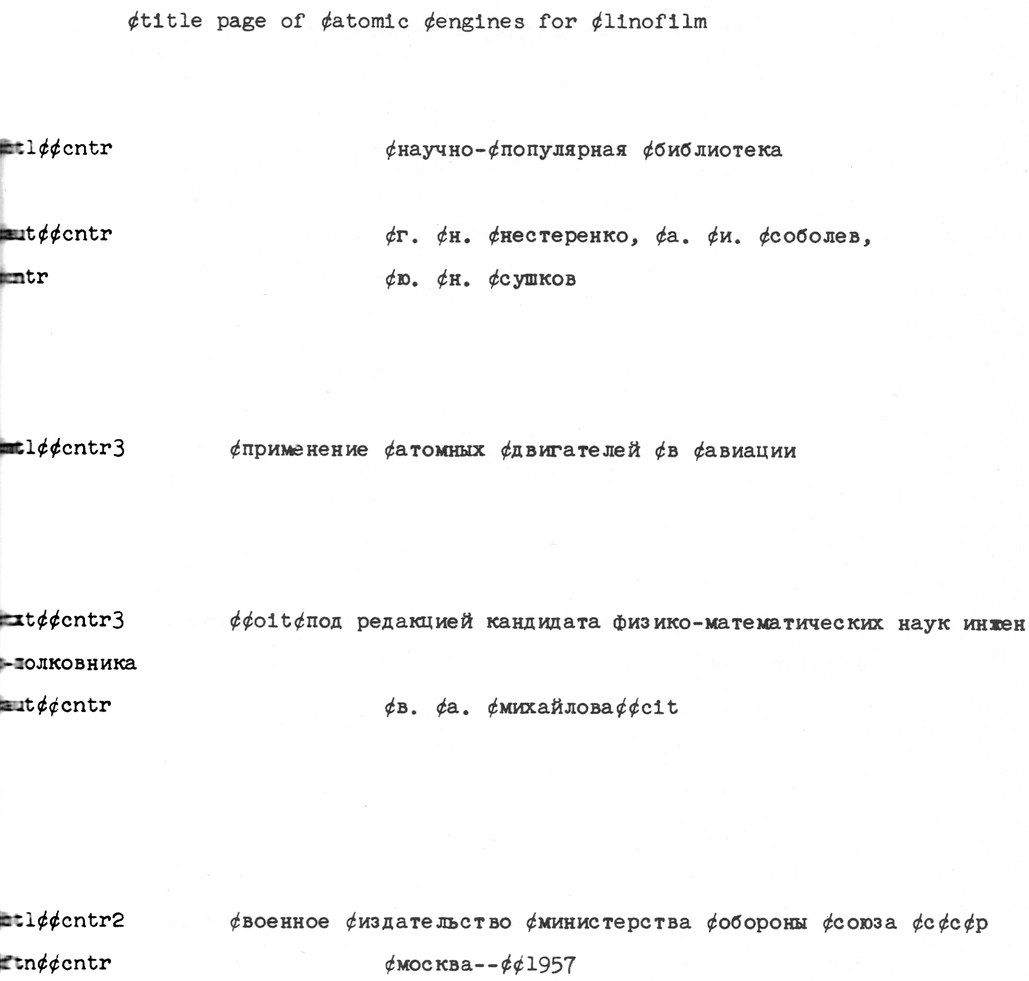
Г. Н. НЕСТЕРЕНКО, А. И. СОБОЛЕВ,
Ю. Н. СУШКОВ

ПРИМЕНЕНИЕ АТОМНЫХ ДВИГАТЕЛЕЙ В АВИАЦИИ

Под редакцией
кандидата физико-математических наук
инженер-полковника
В. А. МИХАЙЛОВА

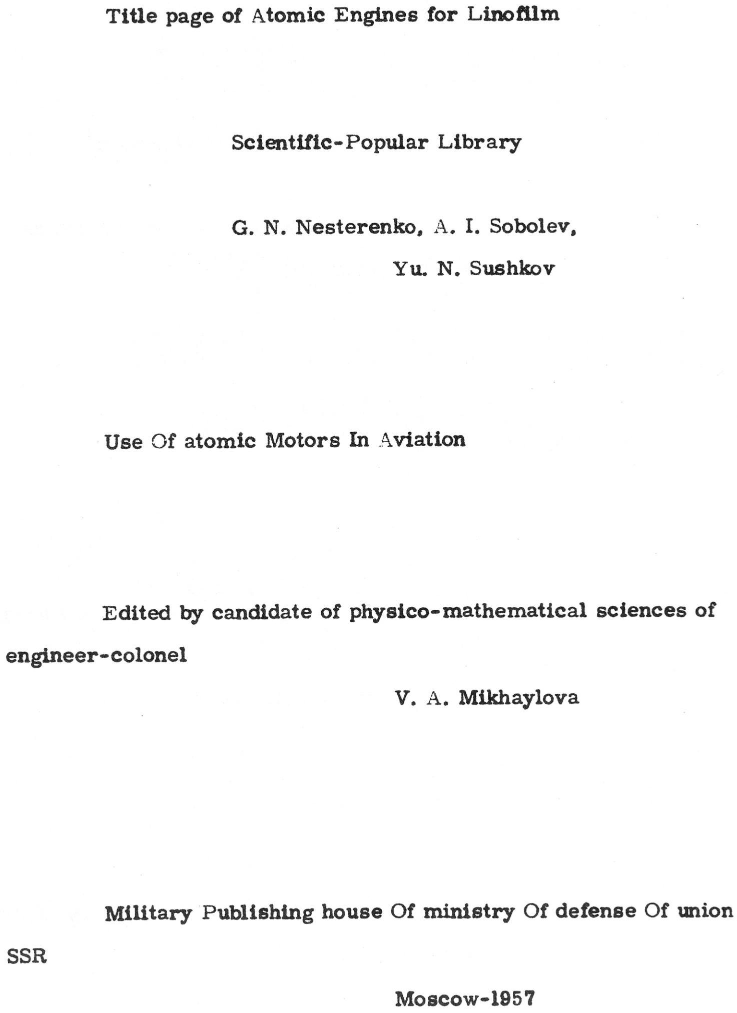
ВОЕННОЕ ИЗДАТЕЛЬСТВО
МИНИСТЕРСТВА ОБОРОНЫ СОЮЗА ССР
Москва — 1957

Example 8.1. *Title page of a Russian document translated and typeset by the IBM Watson Research Center.*[21]



```
        ¢title page of ¢atomic ¢engines for ¢linofilm


l¢¢cntr                      ¢научно-¢популярная ¢библиотека

t¢¢cntr                      ¢г. ¢н. ¢нестеренко, ¢а. ¢и. ¢соболев,
tr                           ¢ю. ¢н. ¢сушков



l¢¢cntr3          ¢применение ¢атомных ¢двигателей ¢в ¢авиации



t¢¢cntr3          ¢¢oit¢под редакцией кандидата физико-математических наук инжен
-олковника
t¢¢cntr                      ¢в. ¢а. ¢михайлова¢¢cit



l¢¢cntr2          ¢военное ¢издательство ¢министерства ¢обороны ¢союза ¢с¢с¢р
n¢¢cntr                      ¢москва--¢¢1957
```

Example 8.2. *Russian language Flexowriter input to the translating computer.*



Title page of Atomic Engines for Linofilm

Scientific-Popular Library

G. N. Nesterenko, A. I. Sobolev,
Yu. N. Sushkov

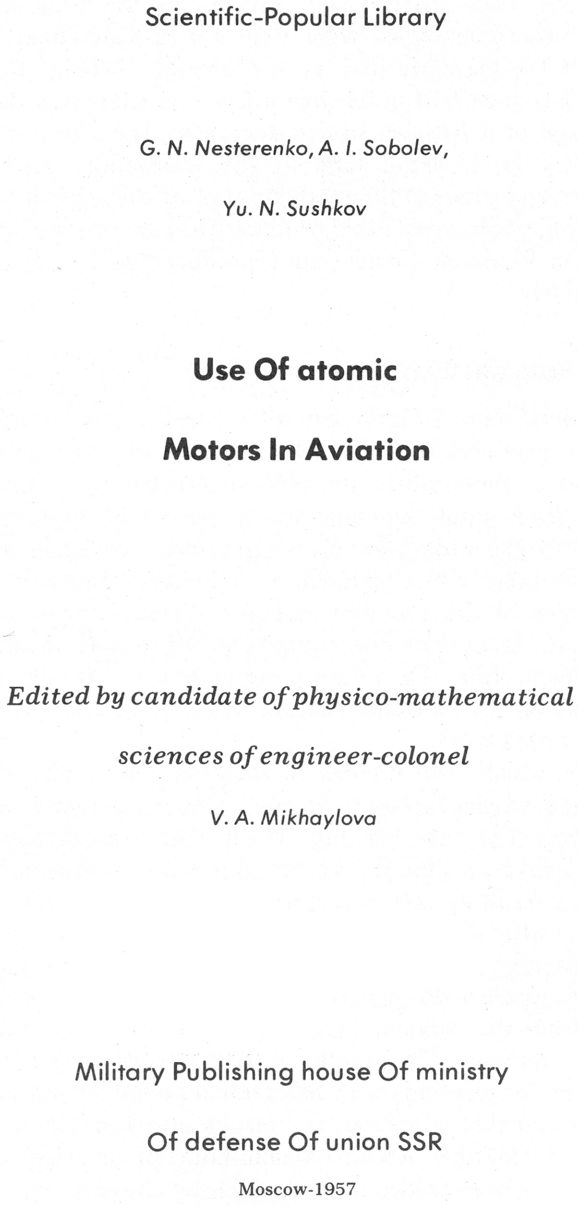
Use Of atomic Motors In Aviation

Edited by candidate of physico-mathematical sciences of
engineer-colonel
V. A. Mikhaylova

Military Publishing house Of ministry Of defense Of union
SSR
Moscow-1957

Example 8.3. *Flexowriter form of the output from the translating computer.*

Title page of Atomic Engines for Linofilm



Scientific-Popular Library

G. N. Nesterenko, A. I. Sobolev,

Yu. N. Sushkov

Use Of atomic

Motors In Aviation

Edited by candidate of physico-mathematical

sciences of engineer-colonel

V. A. Mikhaylova

Military Publishing house Of ministry

Of defense Of union SSR

Moscow-1957

Example 8.4. *Phototypeset product of the computer translating and typesetting processes.*

mnemonics were used to open italics (oit) and to close italics (cit). Further mnemonics were used to center and to flush unjustified matter left and right. Text material was justified by the program. Two- and three-letter mnemonics were used for special characters, for example, th for therefore and gr for greater. Typing these mnemonics for a sequence of more than a few characters was rather irksome. A page of a Russian source document, the Flexowriter copy corresponding to an input tape to the translation machine, the Flexowriter copy produced from the output of this machine, and the corresponding phototypeset end product (these were prepared at the IBM Watson Research Center) are reproduced as Examples 8.1 to 8.4, respectively.

8.3 Table Setting at the NBS

As a result of many of its research activities, the National Bureau of Standards produces reference tables that are often of considerable length. Most of these tables are now constructed by computers, in some cases from input that consists of the results of scientific experiments.[23, 24] The initial computer output that was obtained recently for one such table was converted by a further computer program into the codes of the Linofilm machine. These were written on a magnetic tape to operate the equipment which was mentioned in the preceding section. The entire table of nearly 600 pages was set mechanically on the Linofilm machine in this way, without the need for any keyboard work.

The table, which is of interest to chemists and to physicists who deal with the science of spectroscopy, contained numerical entries in six columns. The page heading, which is uniform throughout the table, was typed on a Linofilm keyboard perforator and the information from the resulting tape was stored in the computer and repeated in the output after each page break.

The typesetting program was very simple. It was written in FORTRAN, apart from two short FAP subroutines. The table was produced from the original input data in a 20-minute run on an IBM 7090 computer. The results were recorded in binary coded decimal form, for printing on a conventional computer output device. The further run that converted the results into Linofilm code took 13 minutes of 7090 time. About 300 man-hours of keyboard operators time would have been needed to set the table by conventional methods. Preparation of the film in the phototypesetting machine took 80 hours. The equipment was considered to be reliable enough for com-

Intensity	Wavelength A	Energy Levels K	gA 10^8/sec	gf	Log gf
78	2626.68	40944 – 79003	0.042	0.0043	−2.36
131	2630.00	39019 – 77031	0.070	0.0073	−2.14
145	2634.93	39019 – 76959	0.078	0.0081	−2.09
41	2645.30	40114 – 77905	0.022	0.0023	−2.63
60	2649.84	40114 – 77841	0.033	0.0034	−2.47
15	2846.48	40944 – 76064	0.0087	0.0011	−2.98
139	2858.22	39019 – 73995	0.080	0.0097	−2.01
185	2858.73	11203 – 46173	0.010	0.0013	−2.89
65	2890.84	44406 – 78988	0.038	0.0048	−2.32
56	2891.64	44544 – 79116	0.033	0.0042	−2.38
11	2931.70	40944 – 75044	0.0065	0.00084	−3.08
29	2933.06	39019 – 73103	0.017	0.0022	−2.66
119	2978.30	43514 – 77080	0.072	0.0096	−2.02
103	2979.38	43514 – 77068	0.063	0.0083	−2.08
98	2998.38	11203 – 44544	0.0036	0.00048	−3.32
161	3012.00	40114 – 73305	0.096	0.013	−1.88
54	3014.85	41153 – 74313	0.033	0.0045	−2.35
191	3021.54	40909 – 73995	0.12	0.016	−1.80
179	3022.61	39019 – 72093	0.11	0.015	−1.84
90	3024.99	30535 – 63585	0.044	0.0060	−2.22

Example 8.5. *A portion of a table phototypeset with computer aid, by the National Bureau of Standards.*[23], [24]

plete proofreading to be unnecessary. Only the first and last entry on each page were checked. Some errors have been found since publication.

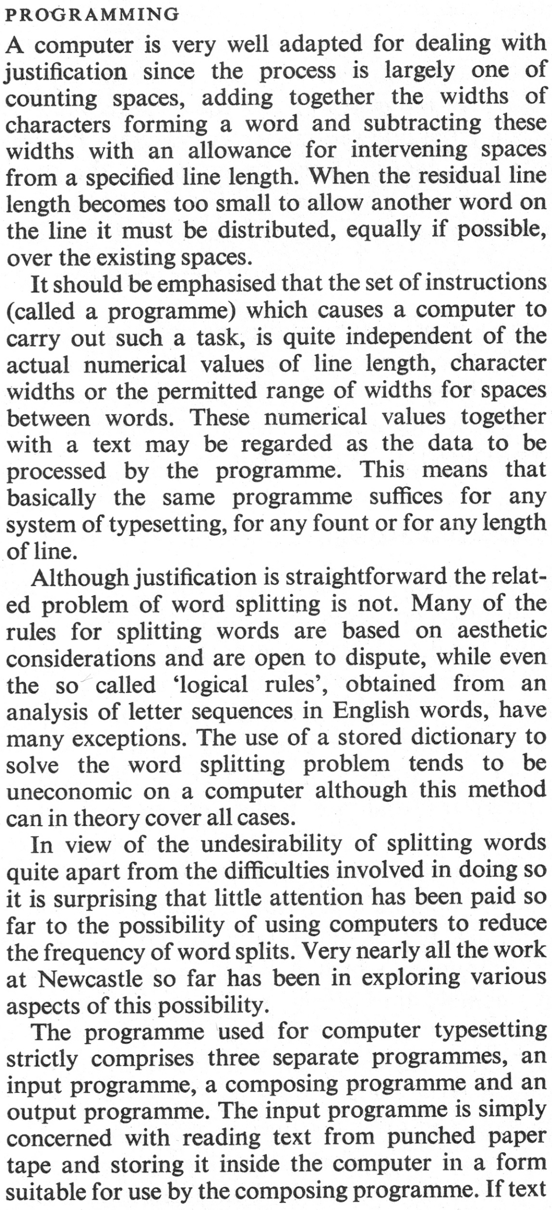
A portion of the typeset table is reproduced as Example 8.5. The table consisted of experimental transition probabilities for the spectral lines of seventy elements.

8.4 The Newcastle Project

A computer program that produces an operating tape for a Monotype caster has been written at Newcastle University.[25] The input to the program is punched on a simple keyboard perforator by an operator, without consideration of line and word breaks in the typeset end product. The computer determines the positions of these breaks and calculates the spaces that must be left between the words to justify the type. The program deals with straight text matter and has been used, for example, to set the text of a 20-page paper that appeared in *Printing Technology*.

Particular attention has been given to the need to hyphenate material, and experiments with the program have confirmed several of the expectations of the Newcastle team. These were that (1) increasing the allowed interword space decreases the need to hyphenate words at line endings, (2) hyphenation is more frequent in matter that has a high incidence of long words, (3) hyphenation is more frequent for matter set in narrow measure. Detailed statistics compiled from computer runs with sample material by three authors are given to support these views.

The Newcastle team uses a justification procedure that is rather more elaborate than that which most other workers have employed. To avoid splitting words at line endings whenever possible, the positions of line breaks within a paragraph are not finalized until the paragraph is completely processed. Whenever an interword space is encountered, the minimum output space is allocated to it tentatively. When overset occurs, the word that is being processed is left for inclusion in the next line. A test is made to determine if the interword spaces on the line that has just been completed can be expanded to fill the requisite measure without exceeding the limit that has been specified for the distance between words in the typeset end product. If the interword space would become excessive, a test is made to determine if the last word on the previous line can be brought down without making the spaces needed to justify that line become excessive. If the spaces would remain within the allowed limit, the word



PROGRAMMING

A computer is very well adapted for dealing with justification since the process is largely one of counting spaces, adding together the widths of characters forming a word and subtracting these widths with an allowance for intervening spaces from a specified line length. When the residual line length becomes too small to allow another word on the line it must be distributed, equally if possible, over the existing spaces.

It should be emphasised that the set of instructions (called a programme) which causes a computer to carry out such a task, is quite independent of the actual numerical values of line length, character widths or the permitted range of widths for spaces between words. These numerical values together with a text may be regarded as the data to be processed by the programme. This means that basically the same programme suffices for any system of typesetting, for any fount or for any length of line.

Although justification is straightforward the related problem of word splitting is not. Many of the rules for splitting words are based on aesthetic considerations and are open to dispute, while even the so called 'logical rules', obtained from an analysis of letter sequences in English words, have many exceptions. The use of a stored dictionary to solve the word splitting problem tends to be uneconomic on a computer although this method can in theory cover all cases.

In view of the undesirability of splitting words quite apart from the difficulties involved in doing so it is surprising that little attention has been paid so far to the possibility of using computers to reduce the frequency of word splits. Very nearly all the work at Newcastle so far has been in exploring various aspects of this possibility.

The programme used for computer typesetting strictly comprises three separate programmes, an input programme, a composing programme and an output programme. The input programme is simply concerned with reading text from punched paper tape and storing it inside the computer in a form suitable for use by the composing programme. If text

Example 8.6. *A product of the Newcastle University computer program for typesetting.*[25]

is brought down and the processing continued from that word. If the spaces would become excessive, the previous lines are tested one by one, going backward, until one is found in which the last word can be brought down to the next line and the residual material expanded without exceeding the allowed interword spacing. Processing is then restarted from that word. If the beginning of the paragraph or a line that ends with a hyphenated word is encountered before a line that ends with a word that can be brought down, the process is abandoned, and the word whose overset initiated the search is hyphenated.

The hyphenation procedure makes use of some simple rules that are described by the Newcastle team in detailed reports which they have published. They claim 65 to 70 per cent success according to some criteria which they apply. A sample of the material which they have set is reproduced as Example 8.6.

8.5 Commercial Text Setting

The aspect of computer-aided typesetting which has attracted most public attention concerns the use of the computer to introduce line and word breaks in text matter that is typed by keyboard operators, who are freed from the burden of determining these breaks as they type.[26] The possible impact of this form of computer-aided typesetting on the newspaper and book production industries has been the subject of considerable comment in the trade and popular press. The programs that are being offered and employed commercially relate primarily to a production process that can be regarded as an extension of the conventional use of keyboard perforators and tape-operated typesetting machines in graphic arts work.

The conventional use of paper tape in typesetting arose in connection with equipment which has not been mentioned previously in this book, namely linecasting machines. To put the equipment that the author used into perspective, the overwhelming bulk of typesetting is performed today on hot metal equipment, which includes Monotype casters which were mentioned in the preceding section, and Linotype and Intertype linecasting machines. The Monotype casters are operated by perforated rolls of paper, called "spools," which are punched on special keyboard perforators. A keyboard that activates the selection of metal type is provided on every linecasting machine. A tape reader that is called an operating unit can also be attached to a linecasting machine to set type under the control of paper tape that is punched by keyboard operators using the TTS perforator or other keyboard perforators with the same tape code.

A substantial amount of linecasting work is tape operated. The keyboard perforators are equipped with mechanical or electrical units that represent tables of character widths. The measure in which material is to be set is registered on a further unit, and as the operator types, a record of the line deficit is maintained by a simple subtracting device. It is indicated by the position of a pointer or in some other way that the operator can see. When the operator decides that a new line must start in the typeset end product, he (or she) presses the end-line key, which punches a code in the tape that will make the linecasting machine start a new line of type. The range of deficits in which the operator may break a line is called the justification zone, and it is determined by the looseness or tightness of interword spacing that is required in the typeset product. Some perforators, such as the TTS and Brewer perforators, do not provide hard copy. Other perforators, such as the LCC perforator, do give hard copy in a simple typewriter face, and execute a carriage return when the operator presses the end-line key to start a new line of cast type.

The paper tapes that operate linecasting machines are punched in a very simple code, as a consequence of the way in which these machines work. Successive characters that are to be set in type are represented by successive codes in the tape. Further codes are used to end lines, change case, and to change font. In general, two fonts in the same point size can be used in a single run and these two fonts can be changed between runs. A line of matrices is assembled mechanically character by character as the tape is read. When a character code is read, the operating unit releases the appropriate matrix from the magazine, and the matrices that are released by the successive codes are assembled in the correct order. When a code is read that represents a space which may be expanded for purposes of justification, a "space band" is dropped into place. This is a piece of metal that closes into a wedge shape on its lower face. When the end-line code is read, the space bands are forced down mechanically until the further outward movement of the matrices is arrested by a metal stop. The line of matrices is brought in contact with a mould, from which the line of type is cast and dropped into a galley. The matrices are returned mechanically to appropriate positions in the magazine for further use. The physical width of the matrices and the physical process by which interword spaces are expanded make it unnecessary to punch width codes in the tape, even though the width values are used to determine line breaks when the tape is punched. Some more elaborate systems of tape operation that permit more flexible control of the spacing are also in use.

The development of phototypesetting equipment has paralleled that of linecasting machines in that the use of manually operated machines became established first, and has been followed by the provision of systems in which a keyboard perforator unit is used to punch tape for a mechanically operated photo unit. In the development of tape-operated phototypesetting machines, the tendency has been to use a tape code that results primarily from the way that the workings of these machines have evolved, which are very different from those of the linecasting equipment. Thus the tape codes for the Linofilm and Photon 540, which are punched by keyboard perforators designed specifically for these machines, are completely different from those that the TTS perforator punches. There is a tendency now to adapt the phototypesetting machines so that they can be operated by tapes which are punched with the codes that operate linecasting machines, but with different functions accruing to some of the codes or combinations of the codes. This is necessary, for example, to control the greater flexibility of type style and size that is provided by the phototypesetting machines.

The commercial interest in computer-aided typesetting has been linked primarily to linecasting machines that can be operated by paper tapes punched on the TTS perforator and on other perforators with the same tape codes. The perforators have been modified so that they do not use magazines of width values to signal approaching line completion. The computer is used essentially to read the tapes that are prepared on the modified perforators, without end-line codes, and to punch operating tapes with the end-line codes included. Since the modified perforators punch the character codes in the precise form that the operating unit of the linecasting machines require, there is no need for code translation by the computer. This is also true when the final tape is used to operate a phototypesetting machine that has been adapted to accept the same tape code. Some code translation is necessary, however, when a typesetting machine with a different operating code is used.

The basic process that is involved in commercial computer-coded text setting is that which the Newcastle group investigated and which has been discussed in the preceding section. The measure is a program constant, or a parameter that is read at the beginning of the input. Line breaks are determined by reference to a table of width values, and where the interword spaces needed to justify a line by expansion are excessive, the word that causes overset is hyphenated. Some programs hyphenate by a table-search process, some by reference to a set of rules, and some by a combination of these methods. None give perfect hyphenation as yet. The syntactic analysis methods that are discussed in Chapter 7 seem relevant to this problem. Some commercial programs have editing facilities, and corrections can always be made manually on linecasting machines. The programs that are in use allow some typographic options such as indents, hanging indents, and font to be specified by the keyboard operator.

The commercial interest in these programs has been strongest in the newspaper industry, which deals with short "takes" (items of text). It is reported that several IBM 1620's, RCA 301's, and an NCR 315 have been installed for newspaper and for some book work in the United States. Some attention is being given to the use of wire service material, combined with local amendations, as input to the process.

8.6 Man-Machine Interaction

During the early years, the scene that was perhaps most characteristic of the realities of high-speed computing contained a programmer or operator, displaying emotions that ranged from the desperate to the euphoric, watching output being printed on a tabulator, and instructing the computer how to proceed, by pushing buttons, throwing switches, or pressing keys. Then it became demodé for several years for users to get anywhere near the larger computers, which were run under the relentless control of monitor programs and inflexible regulations that precluded rapid feedback between the computer and its user. Man-machine interaction has come back into vogue, however, and the objective of much research and expenditure in computer science today is intended to facilitate work in which the computer pauses in its execution of a job so that a human supervisor may specify the course of action which should be taken, at points in a process which are better left to human judgment than to programmed rules.

This approach of "on-line" interaction with a computer is being adopted in some of the editing work which is mentioned in Section 8.8. It is being used to good advantage in a special-purpose computing device that determines line breaks mechanically wherever this does not require hyphenation, but allows an operator to hyphenate when a word must be broken. The Linosec machine, which works in this manner,[27] produces operating tape for linecasting machines (and for phototypesetting machines that can be operated by the same tape codes) from an input tape that is punched on a keyboard perforator without consideration for line and word breaks in the typeset end product. The measure that is to be used is specified by a dial setting, and width values for a type face are provided in a small pluggable unit that is changed for different fonts. Loose or tight spacing can be specified by a switch on the console.

When a word must be broken to justify a line without exceeding the allowed interword space, the portion of the word up to the character causing overset is displayed on a cathode-ray tube. The operator presses a button under the letter which should be followed by a hyphen, and the machine continues to operate automatically until another word is encountered that requires hyphenation.

An input word that is followed by a "word-delete" code is eliminated by the Linosec from the output. This deals with errors that the keyboard operator recognizes as soon as they are made. Indents, hanging indents, and other typographic variations can be introduced in the output tape by suitable settings of the Linosec controls.

A second model of the Linosec performs the same functions on tapes that can be read from 12 input stations, and punches justified tapes at 12 output stations, so that it can deal with the work of several keyboard operators and linecasting machines concurrently.

8.7 The Bibliographic Reference Literature

The effectiveness with which medicine, law, and other professions can be practiced, and research and development conducted in many scientific and other fields, depends increasingly on the availability of extensive bibliographic services. The explosion of activity in many fields of endeavor requires the use of highly mechanized procedures to deal with the practical problems that arise in the compilation and publication of bibliographic works.

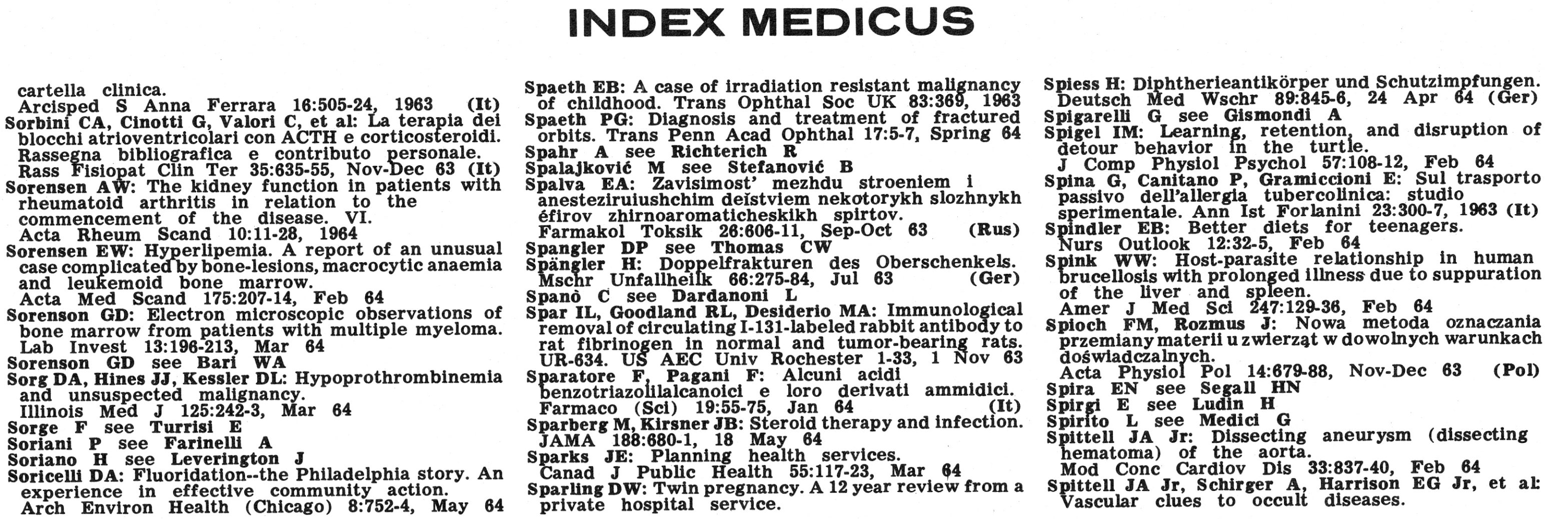
The MEDLARS project of the National Library of Medicine, in the United States, has begun recently to use a fairly powerful computer, and a high-speed phototypesetting machine that is operated by magnetic tape, to produce the bibliographic publication *Index Medicus*.[28] This publication has been produced by other methods since 1879. In recent years, several stages of mechanization have been introduced, and these have led logically to the introduction of a computer at a critical point in the production process.

The *Index Medicus* is published monthly, and a consolidated issue is published annually. Each monthly issue lists the titles, authors, and citations of all the learned articles of relevance to medical practice and research that have been recorded by the indexing staff of the National Library of Medicine since the production of the previous issue. The indexing staff receives more than 2500 biomedical journals from many different countries. On average, nearly 1500 journal issues arrive each month. This number will probably increase by 50 per cent in the next five years. Each journal contains on average ten articles that are indexed.

The references to journal articles are listed in the subject section of the *Index Medicus* under medical subject headings which indicate the topics to which the articles relate. The headings are arranged in alphabetic order. The indexers only use headings that are included in a list which has been prepared by the N.L.M. staff. An article may be mentioned under several headings. This number is expected to average about three in the computer-generated issues. It has been kept to about two in previous issues. One of the benefits of the present computer process is the ease with which entries can be repeated under different subject headings. References to journal articles are also listed after author names in an alphabetic author index.

Portions of a recent issue of the *Index Medicus* are reproduced as Examples 8.7 and 8.8. In the Subject Section, the medical subject headings in 10-point type are followed in parentheses by codes, such as C2, C4 after esophageal neoplasms, which refer to the position of the subject heading in a categorized list that is issued by N.L.M. The information concerning each journal article is set in 6-point type. It begins with the title if it is written in English, and an English translation of the title in square brackets if it is written in some other language. The name of the author, or first author followed by "et al." if there are several, follows, then the abbreviated journal name, volume number, pagination, and date of issue. If the article is not in English, an abbreviation of the language in which it is written is given in parentheses. In the author index, the author names are printed in boldface for each article, followed by the title in the language in which it is written, and the citation. In some cases cross references are given to co-authors.

Material that is to be included in the *Index Medicus* is typed, together with certain other information about the journal articles, on special forms, by the indexers. After these forms have been checked and corrected if necessary, they are given to Flexowriter operators who retype the information that is on the forms and also type further information from the journal that the indexers have specified. The hard copy that these operators type is proofread, and if it contains errors a list of corrections is typed on the Flexowriter. The Flexowriter tapes form the main input to the computer operations that follow. These produce the magnetic operating tape from which the



INDEX MEDICUS

cartella clinica. Arcisped S Anna Ferrara 16:505-24, 1963 (It)

Sorbini CA, Cinotti G, Valori C, et al: La terapia dei blocchi atrioventricolari con ACTH e corticosteroidi. Rassegna bibliografica e contributo personale. Rass Fisiopat Clin Ter 35:635-55, Nov-Dec 63 (It)

Sorensen AW: The kidney function in patients with rheumatoid arthritis in relation to the commencement of the disease. VI. Acta Rheum Scand 10:11-28, 1964

Sorensen EW: Hyperlipemia. A report of an unusual case complicated by bone-lesions, macrocytic anaemia and leukemoid bone marrow. Acta Med Scand 175:207-14, Feb 64

Sorenson GD: Electron microscopic observations of bone marrow from patients with multiple myeloma. Lab Invest 13:196-213, Mar 64

Sorenson GD see **Bari WA**

Sorg DA, Hines JJ, Kessler DL: Hypoprothrombinemia and unsuspected malignancy. Illinois Med J 125:242-3, Mar 64

Sorge F see **Turrisi E**

Soriani P see **Farinelli A**

Soriano H see **Leverington J**

Soricelli DA: Fluoridation--the Philadelphia story. An experience in effective community action. Arch Environ Health (Chicago) 8:752-4, May 64

Spaeth EB: A case of irradiation resistant malignancy of childhood. Trans Ophthal Soc UK 83:369, 1963

Spaeth PG: Diagnosis and treatment of fractured orbits. Trans Penn Acad Ophthal 17:5-7, Spring 64

Spahr A see **Richterich R**

Spalajković M see **Stefanović B**

Spalva EA: Zavisimost' mezhdu stroeniem i anesteziruiushchim deĭstviem nekotorykh slozhnykh éfirov zhirnoaromaticheskikh spirtov. Farmakol Toksik 26:606-11, Sep-Oct 63 (Rus)

Spangler DP see **Thomas CW**

Spängler H: Doppelfrakturen des Oberschenkels. Mschr Unfallheilk 66:275-84, Jul 63 (Ger)

Spanò C see **Dardanoni L**

Spar IL, Goodland RL, Desiderio MA: Immunological removal of circulating I-131-labeled rabbit antibody to rat fibrinogen in normal and tumor-bearing rats. UR-634. US AEC Univ Rochester 1-33, 1 Nov 63

Sparatore F, Pagani F: Alcuni acidi benzotriazolilalcanoici e loro derivati ammidici. Farmaco (Sci) 19:55-75, Jan 64 (It)

Sparberg M, Kirsner JB: Steroid therapy and infection. JAMA 188:680-1, 18 May 64

Sparks JE: Planning health services. Canad J Public Health 55:117-23, Mar 64

Sparling DW: Twin pregnancy. A 12 year review from a private hospital service.

Spiess H: Diphtherieantikörper und Schutzimpfungen. Deutsch Med Wschr 89:845-6, 24 Apr 64 (Ger)

Spigarelli G see **Gismondi A**

Spigel IM: Learning, retention, and disruption of detour behavior in the turtle. J Comp Physiol Psychol 57:108-12, Feb 64

Spina G, Canitano P, Gramiccioni E: Sul trasporto passivo dell'allergia tubercolinica: studio sperimentale. Ann Ist Forlanini 23:300-7, 1963 (It)

Spindler EB: Better diets for teenagers. Nurs Outlook 12:32-5, Feb 64

Spink WW: Host-parasite relationship in human brucellosis with prolonged illness due to suppuration of the liver and spleen. Amer J Med Sci 247:129-36, Feb 64

Spioch FM, Rozmus J: Nowa metoda oznaczania przemiany materii u zwierząt w dowolnych warunkach doświadczalnych. Acta Physiol Pol 14:679-88, Nov-Dec 63 (Pol)

Spira EN see **Segall HN**

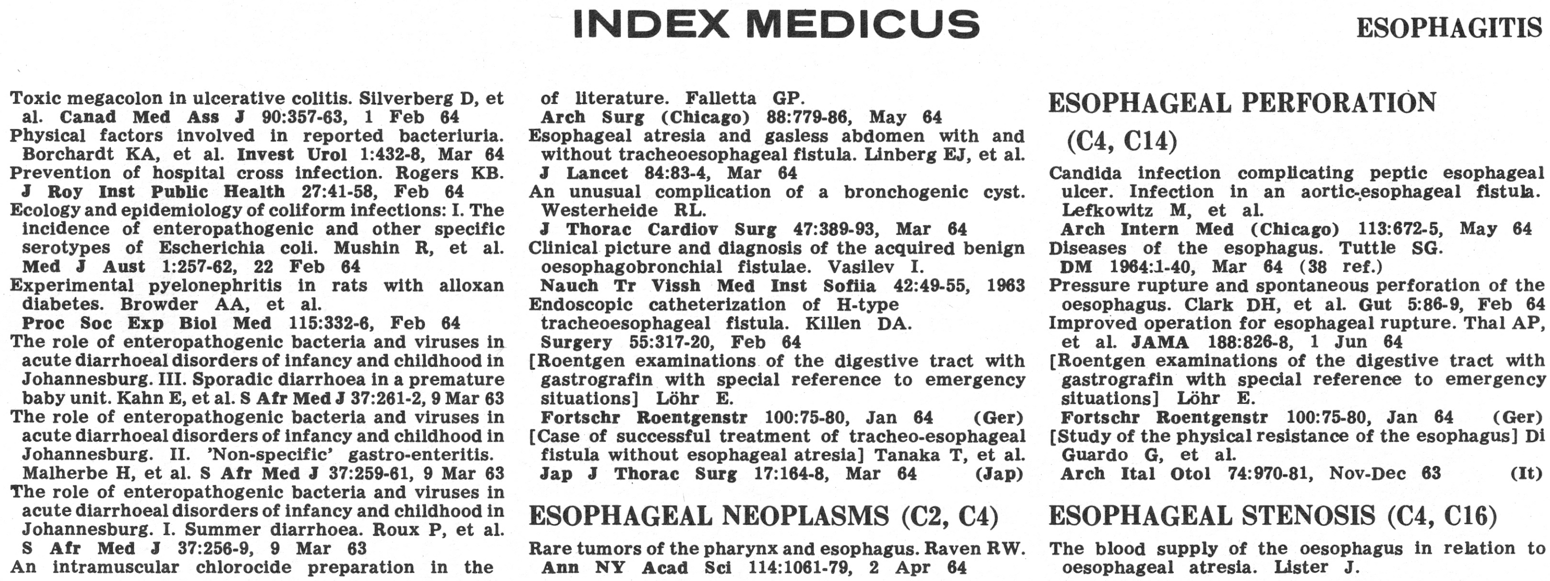
Spirgi E see **Ludin H**

Spirito L see **Medici G**

Spittell JA Jr: Dissecting aneurysm (dissecting hematoma) of the aorta. Mod Conc Cardiov Dis 33:837-40, Feb 64

Spittell JA Jr, Schirger A, Harrison EG Jr, et al: Vascular clues to occult diseases.

Example 8.7. *Computer-aided phototypeset text produced by the National Library of Medicine (MEDLARS Project).*[28]



INDEX MEDICUS ESOPHAGITIS

Toxic megacolon in ulcerative colitis. Silverberg D, et al. **Canad Med Ass J** 90:357-63, 1 Feb 64

Physical factors involved in reported bacteriuria. Borchardt KA, et al. **Invest Urol** 1:432-8, Mar 64

Prevention of hospital cross infection. Rogers KB. **J Roy Inst Public Health** 27:41-58, Feb 64

Ecology and epidemiology of coliform infections: I. The incidence of enteropathogenic and other specific serotypes of Escherichia coli. Mushin R, et al. **Med J Aust** 1:257-62, 22 Feb 64

Experimental pyelonephritis in rats with alloxan diabetes. Browder AA, et al. **Proc Soc Exp Biol Med** 115:332-6, Feb 64

The role of enteropathogenic bacteria and viruses in acute diarrhoeal disorders of infancy and childhood in Johannesburg. III. Sporadic diarrhoea in a premature baby unit. Kahn E, et al. **S Afr Med J** 37:261-2, 9 Mar 63

The role of enteropathogenic bacteria and viruses in acute diarrhoeal disorders of infancy and childhood in Johannesburg. II. 'Non-specific' gastro-enteritis. Malherbe H, et al. **S Afr Med J** 37:259-61, 9 Mar 63

The role of enteropathogenic bacteria and viruses in acute diarrhoeal disorders of infancy and childhood in Johannesburg. I. Summer diarrhoea. Roux P, et al. **S Afr Med J** 37:256-9, 9 Mar 63

An intramuscular chlorocide preparation in the treatment of literature. Falletta GP. **Arch Surg (Chicago)** 88:779-86, May 64

Esophageal atresia and gasless abdomen with and without tracheoesophageal fistula. Linberg EJ, et al. **J Lancet** 84:83-4, Mar 64

An unusual complication of a bronchogenic cyst. Westerheide RL. **J Thorac Cardiov Surg** 47:389-93, Mar 64

Clinical picture and diagnosis of the acquired benign oesophagobronchial fistulae. Vasilev I. **Nauch Tr Vissh Med Inst Sofiia** 42:49-55, 1963

Endoscopic catheterization of H-type tracheoesophageal fistula. Killen DA. **Surgery** 55:317-20, Feb 64

[Roentgen examinations of the digestive tract with gastrografin with special reference to emergency situations] Löhr E. **Fortschr Roentgenstr** 100:75-80, Jan 64 (Ger)

[Case of successful treatment of tracheo-esophageal fistula without esophageal atresia] Tanaka T, et al. **Jap J Thorac Surg** 17:164-8, Mar 64 (Jap)

ESOPHAGEAL NEOPLASMS (C2, C4)

Rare tumors of the pharynx and esophagus. Raven RW. **Ann NY Acad Sci** 114:1061-79, 2 Apr 64

ESOPHAGEAL PERFORATION (C4, C14)

Candida infection complicating peptic esophageal ulcer. Infection in an aortic-esophageal fistula. Lefkowitz M, et al. **Arch Intern Med (Chicago)** 113:672-5, May 64

Diseases of the esophagus. Tuttle SG. **DM** 1964:1-40, Mar 64 (38 ref.)

Pressure rupture and spontaneous perforation of the oesophagus. Clark DH, et al. **Gut** 5:86-9, Feb 64

Improved operation for esophageal rupture. Thal AP, et al. **JAMA** 188:826-8, 1 Jun 64

[Roentgen examinations of the digestive tract with gastrografin with special reference to emergency situations] Löhr E. **Fortschr Roentgenstr** 100:75-80, Jan 64 (Ger)

[Study of the physical resistance of the esophagus] Di Guardo G, et al. **Arch Ital Otol** 74:970-81, Nov-Dec 63 (It)

ESOPHAGEAL STENOSIS (C4, C16)

The blood supply of the oesophagus in relation to oesophageal atresia. Lister J.

Example 8.8 *Another sample of the MEDLARS Project output.*

monthly *Index Medicus* is set on the GRACE machine, which is described below. Files of information are also produced on magnetic tape for certain further purposes, which are discussed at the end of this section. Examples 8.7 and 8.8 are copied from a recent issue of the *Index Medicus* that was set by these methods.

The keyboard work in the process that has just been described is simpler than would be necessary if the pages of the *Index Medicus* were typed directly. There are several reasons for this. When the Flexowriter operators deal with a journal issue, the journal name and date are typed once, and the information concerning the articles in that issue are typed in sequence. This information, which is typed only once, is repeated several times in the published *Index Medicus*. The material that is collected for an issue of the *Index*, moreover, can be put into the correct sequence at high speed by the computer when it is complete. This and the high speed of the typesetting machine enable an issue to be published very soon after the collection of the information which it contains is complete.

The retention of material in the monthly *Index Medicus* on magnetic tape will make it possible to produce the corresponding magnetic tapes for the annual issue by a merging process on the computer, which simplifies the production of the annual issue considerably.

In addition to expediting the production of the *Index Medicus* as just described, the MEDLARS project will provide further benefits of considerable magnitude. The indexers who record the journal articles type many items of information about these articles besides the items that appear in the *Index Medicus*. These are all punched by the Flexowriter operators, and carried through the computer processes that organize the information in files on the magnetic tape. The program which produces the operating tape for the published *Index Medicus* on GRACE only includes the items that are to be set in type. The further material is still available on the main reference tapes, however, for mechanical searching by the computer, to produce specialist bibliographies. A major objective of the MEDLARS project is the production of many such bibliographies, both of a recurrent variety and in answer to individual requests for information on particular topics. It is estimated that about fifty specialist bibliographies will be issued on a recurrent basis with periodicities that will range from one week to six months. It is expected that by 1969 over twenty thousand nonrecurrent inquiries will be answered each year.

The GRACE machine, the phototypesetting machine that is used to set the *Index Medicus*, contains a glass grid on which are the negatives of about 230 characters. A separate flash tube is mounted behind each of these negatives. A lens travels at high speed back and forth along a path that is parallel to the grid of negatives. A horizontal strip of a 100-foot roll of film is exposed by a window that is parallel to the path of the lens. If a character on the grid is illuminated by its flash tube, an image is formed on the exposed strip of film. The position of the image depends on the position of the character in the grid and the position of the lens at the instant of illumination. The image of any character can be formed at any point along the film by illuminating the character at the appropriate instant. This is when the lens crosses the line that joins the character in the grid to the point on the film where it must be formed. The GRACE machine contains electronic units that read and store the computer codes which represent a line of type on the magnetic tape. From this sequence of codes, the circuits determine when the individual characters in the grid should be illuminated as the lens moves along its path. The tubes behind the grid are flashed accordingly, and the film is advanced mechanically each time the lens reverses its direction of movement. The machine sets about 300 characters per second.

8.8 Electron-Tube Devices

In all the phototypesetting machines which have been mentioned so far in this book, operating speeds and other characteristics are limited essentially by the time that lenses, prisms, and other objects take to move. These movements select the characters which are projected and also determine the positions at which the images of these characters are formed on the film. During the last few decades, an extensive technology of electron-tube devices has developed, which perhaps is best known in the context of television. The impact of a beam of electrons on certain materials causes them to glow in the region of impact and for the duration of impact. The path of a beam of electrons can be determined by the relative strengths of the electric and magnetic fields through which it moves. Since these can be altered at very high speed by suitable electronic circuitry, the path of an electron beam can be changed very rapidly. In an electron-tube device, the electrons that are emitted from a cathode at one end of the tube pass through electric and magnetic fields which focus it, and then fall on the inner surface of a thin coating of phosphorescent material at the other end of the tube. The portion of this material that is bombarded by electrons glows, and this glow is visible from the outside when the tube is viewed end-on.

In the simplest form of electron tube, a narrow beam of electrons produces a point of light wherever it falls on the end of the tube. Successive points of light are created in this way as the electron beam is swept across the tube and switched on and off as necessary. This is the basis of the television tube. It also is the basis of the cathode-ray tube attachments of digital computers. Extensive use is made of such attachments to form graphs and diagrams that are made up of points whose positions have been calculated by a computer. Characters can be formed in a similar manner as clusters of points at positions on the screen which have been computed. It is the practice to label graphical computer output on a cathode-ray tube by sequences of such characters. The typographic quality in general is very low, but the material is readable on the screens which are viewed directly, and on photographs which are taken of the displays. Computers with attached cathode-ray tubes are being used for experiments in mechanized text editing. The objective of such experiments is to feed a source text into the computer on tape or cards and to display it on the screen, page by page. Corrections and changes are typed on a keyboard that is also attached to the computer. In some cases, the positions at which changes are to be made are indicated by pointing an electronic device called a light pen at the appropriate point on the screen.

The possible use of a more elaborate form of electron-tube device in graphic arts work has been mooted for several years and will be achieved soon with the Linotron.[29] If a wide beam of electrons with parallel paths is fired at an aperture, the emergent beam retains the outline of the aperture. Suitably controlled electric and magnetic fields can deflect this beam without any distortion of its outline. This principle is used in radar displays to provide easily recognized labels that are associated with the spots that represent different sorts of aircraft. The same principle is used in the Charactron device,[30] and more recently in equipment such as the SC 4020,[31] which is equipped with a small range of characters that can be projected in very rapid succession, and which is controlled by computer-generated magnetic tape. This latter machine photographs successive pages of output on successive frames of 16-mm film. Enlargements of these can be printed if necessary. The SC 4020 can be used as a high-speed output device, whose typographic quality is low by graphic arts standards. It has also been used to produce some very effective cinematographic diagrams.[32]

A very high standard of typographic quality, consequent on the use of precise focusing fields in the electron tube, is expected in the Linotron equipment, which will be installed at the U.S. Government

Printing Office and elsewhere. Characters will be provided on grids that can be inserted in the electron tube. Each grid will carry 256 characters in a single point size. It will be possible to use up to four grids in the tube concurrently. The equipment will be operated by magnetic tape, on which a computer has written codes (1) to select a grid whenever this changes, (2) to select characters on the grid in use, (3) to position selected images of these on the screen, (4) to select the magnification to give any of eight point sizes in the range five to eighteen and (5) to advance the film frame in the camera that photographs the screen. It will be possible to project the characters in any sequence, which will make it relatively easy to write programs to generate codes for built-up formulas and for multiple-column pages. It will be possible to produce output in normal page form, or in turned page mode, or without page breaks. The expected speed is in excess of 1000 characters per second in 6-point type, with a higher speed for proof material of lower quality.

The Linotron will be used in the Government Printing Office to set computer-generated material of both a nonsentential and a sentential nature. Very extensive tables and catalogs are produced on computers by many U.S. government departments. An increasing use is being made of the computer, too, in these departments to update text material and to organize and search it for purposes of information retrieval, as for example in the Defense Documentation Agency.

8.9 Other Projects

Several other projects that involve computer-aided typesetting have been reported. The U.S. Geological Survey now prepares a recurrent bibliography for use within the Survey by a computer-aided typesetting process. Source material is keyboarded on a Flexowriter with precedence codes for font, size, and special characters. A Burroughs B 283 computer performs the justification and code translation operations to produce an operating tape for a Photon 200 machine equipped with a 6-channel tape reader and "automated set device." The computer is used to sort the bibliographic items, which are presented in an arbitrary order, into the correct sequence. Indexes are typed separately and processed in a similar manner. Text matter is set using five fonts, subscripts, superscripts, and accents. The publication is one of the few examples of a system that is actually in use and benefiting working scientists.

Several other U.S. government agencies are investigating the mechanized production of a variety of publications in anticipation of the availability of the Linotron machine. In some cases typeset documents will be the end products, in other cases these will be by-products of the input to a mechanized information storage and retrieval system.

The possible publication of scientific journals by a computer-aided process is being studied by Inforonics Corporation. The study is intended to show that the manuscript of a journal article can be typed using appropriate conventions, to produce a paper tape that serves both as input to a computer-aided typesetting process and also to a storage and retrieval system.

Simplified methods of setting built-up chemical and mathematical formulas are under investigation. A special keyboard perforator has been developed at Walter Reed Hospital to punch a representation of a formula in a paper tape when the formula is typed in a fairly simple manner. Programs are being written to produce an operating tape to set formulas on a Photon machine, from a tape that has been punched on this keyboard perforator.

Text editing by computer is being studied by several organizations as a way to update technical reports and manuals. A working system of this nature is used by IBM Corporation to produce successive editions of instruction manuals and other reference literature. Magnetic tape-operated 1403 line printers are used to "print" the publications on lithographic paper plates from which bulk supplies are reproduced. The 1403 provides upper- and lower-case characters but does not allow the spaces to be varied to justify the product.

The IBM editing system has been mentioned in this section, even though it does not use graphic arts typesetting equipment, since text editing is a rather central problem in the effective use of computer-aided typesetting to deal with sentential material. There is no sharp boundary, however, between this type of activity, which can be regarded as a mechanization of work that is primarily publication, and the multitude of computer applications which produce material that must be disseminated more or less widely, and which provide an incentive to the use of tape-operated typesetting machines to set the end products. Some further comments on the variety of potential applications of computer-aided typesetting are provided in Chapter 11, which relates developments in this field to the general progress of information science.

9. The Economics of Computer-Aided Typesetting

9.1 Introduction

Several examples of computer-aided typesetting processes that were studied in the C.C.L. and elsewhere have been reviewed in the preceding chapters of this book. In the present chapter, a wider range of possible applications of computers to typesetting is categorized in Section 9.2. The factors that may affect the economics of such applications are listed in Section 9.3. Some of these factors are discussed in further detail in Sections 9.4 to 9.8, which deal respectively with programming costs, equipment costs, over-all production speeds, keyboarding practice, and the design of typesetting machines.

9.2 Computer-Aided Typsetting Processes

The general applications of computer-aided typesetting processes can be categorized in terms of the stages of operation as it has been described.

Input Preparation Stage.

In this stage, input tapes and decks of cards that contain matter to be set in type and specifications of typographic style and format can be obtained from one or more of the following sources:

1. *Keyboard perforators* with typewriter and other keyboard ar-

rangements that are operated by keyboard operators.

2. *Scientific measuring instruments* with attachments which convert dial readings, such as voltages, into numbers, and punch codes for these in tape or cards, bypassing manual transcription and keyboarding.

3. *Punched-card and tape-handling machines and*

4. *Computers* that read input punched on cards or tapes by any method mentioned here, perform logical and arithmetic processes on this input for a scientific or commercial application, and punch the results in code on cards or tapes.

5. *Communications systems*, e.g., teletype, operated manually by keyboard telegraphists, or transmitting material already punched in cards or tapes at the transmitting station by any process, 1 to 4.

6. *Standing files* of tapes or cards punched previously by any process, 1 to 5.

Organization Process Stage.

In these the computer selects, repeats, and rearranges portions of the input and punches these in the output tape that is then used to operate the typesetting machine. Code translation may precede or follow the organization processes in the computer. Examples of organization processes include the following:

1. *Extraction.* The computer can be given text matter and coded instructions, in punched cards or tape. The instructions can make the computer punch an output tape that reproduces only selected portions of the text matter in the input. Thus the computer could read an alphabetic library catalog on tape in which the date of publication of each book was punched after its title, and then punch a tape to set an alphabetic catalog of the books that were published in a given year.

2. *Repetition.* The computer can repeat one portion of the input in several places in the output. Thus the computer may repeat the description of a book (author, title, publisher) mechanically under several headings in a subject catalog, although keyboard operators punched the description of the book only once.

3. *Sorting.* The computer can be used to put portions of the input into alphabetic or numerical order to form a catalog or index.

4. *Merging.* Several files of text punched on cards or tapes can be merged by the computer into one consolidated file. Thus catalogs for several branch libraries could be merged to form a regional library catalog.

5. *Revision.* Texts, catalogs, indexes, tables, and other works

punched on cards or tapes can be updated by the computer to correct errors and incorporate new information. The computer can make changes at positions which are

a. specified numerically by the operator, for example by counting lines down a page and words across a line, and punching these counts in input cards or tapes.

b. determined mechanically by reference to the literal content of new material that is to be added (e.g., when it is an item to be inserted alphabetically in an index) or old material that is to be replaced (e.g., when the spelling of a particular word must be changed throughout a text).

These processes can be frequently combined. Sometimes process 5 can be used to add or change typographic control information in text matter prior to code translation.

Code Translation and Insertion Stage.

Translation of input codes and code insertion may be necessary for several reasons:

1. Input may be punched by measuring instruments, computers, punched-card or tape-handling equipment, or communication systems without function codes for typographic style and format, or with function and/or character codes that differ from the codes of the typesetting machine. Using a computer, then, for code insertion and/or translation saves operators keyboarding material to be set in type.
2. The input may be punched on a keyboard perforator, and the perforator and typesetting machine may have different codes for reasons of training, capital investment, operating costs, speed, reliability, and standardization.
3. A tape may be punched already for one typesetting machine, and the job must be reset on a different typesetting machine which is adequate for the job but which has different codes.
4. Tapes or cards may be punched by keyboard operators in the code that a computer or a punched-card or tape-handling machine can read, for processing in a nontypesetting application, and the keyboarded material must be set in type as a by-product, without further keyboard operations.
5. The input may be punched by operators whose keyboard work is simplified and therefore faster than in conventional operations because the operator does not determine line endings or hyphenated word breaks.
6. The input may be punched by operators whose work is simplified

since initial type face, type size, body, measure, indention, page depth, folios, runarounds, and other details of typographic control may be specified at the beginning of a text in suitably coded form, and these details changed by the computer, by reference to the following:

a. The literal content of the text or the punctuation, in material that was keyboarded without explicit reference to typography.

b. The format of the keyboarded input.

c. Characters or sequences of characters in the input, that control typography whenever they appear or when they are typed after or between distinctive codes such as brackets, in distinctive locations such as the margin, or in a color shift mode.

d. Codes which are punched by nonprinting keys on the keyboard perforator, that do not affect the appearance of the hard copy.

e. Positions which the operator specified at the beginning of the text, for example by counting lines and words in tabular material.

7. Built-up formulas may be keyboarded in linearized or other simple styles from which the computer forms the codes to set these formulas in type.

8. Complicated page make-up may be described to the computer by words, coordinates, and diagrams, punched in code on tape or cards by a keyboard perforator, and included in the input with tapes for the material to be made up. Text to be set in different columns is merged in the computer to produce a single tape for continuous left-to-right setting.

In some CATP's, a keyboard that is joined by an electrical connection to the computer is used by a keyboard operator to type the text, data, and instructions from which the computer forms the codes that operate the typesetting machine. This is called on-line input operation. It bypasses the need for punched paper tape or cards to be perforated by a punch attachment to the keyboard and to be read by a sensing attachment to the computer.

In some future CATP's, a typesetting machine may be operated by signals which a computer will transmit directly via an electrical connection. This will be called on-line output operation. The operating signals will be formed by the computer from input that is provided in the ways that have been described already. On-line output operation would bypass the need for the computer to punch paper tape or write magnetic tape as an intermediate which is then read by an operating attachment to the typesetting machine.

The computers which are used in CATP's may be either general-purpose computers which are manufactured for use in a diversity of applications, or special-purpose computers that are manufactured for use in CATP's alone.

When on-line input and/or output operation is adopted, several keyboards and/or typesetting machines could be joined to certain computers and used concurrently. This is called time sharing. On-line input and/or output operations may be conducted via lengthy connections between certain computers and keyboards and/or the typesetting machines. This is called remote operation.

9.3 Assessment of Benefits

Computer-aided typesetting may bring benefits where it reduces the time and/or cost required to set a job in type and to print it. If there is a net saving in cost, CATP's may lower the price of publications and increase wages and profits, and make it possible to set jobs that would not otherwise be set by a particular organization, or at all. CATP's would then provide financial, technical, and social benefits to the readers and to other people who are helped by the recipients' acquisition of printed matter, and to bookshops, publishers, printers, and typesetters, and to the suppliers of equipment and materials. If there is a net saving in time, there may be advantages to the users and producers of the printed material, including more up-to-date information for the readers (and attendant benefits to those who are affected by the readers' current awareness) and shorter hours for the typesetters. If CATP's set material which was constructed by a computer and which would otherwise be distributed by photographic reproduction of conventionally typed computer output, then the readability and the usefulness of the material is increased by the availability of more characters, better fonts, and full typographic flexibility, and there is a saving in paper and printing costs and in storage space per copy. Computer-aided processes may allow staff shorter training periods and greater uniformity of operation for varied equipment.

Of course the previous paragraph is largely tautological. The question is whether the computer can provide a net reduction in effective time or cost relative to conventional typesetting in an individual job. Several separate processes and pieces of equipment are involved in a CATP. When analyzing the possible advantages of establishing a facility for CATP's and of using such a facility, each stage of the process must be assessed by reference to numerous criteria and the results of these assessments summed over all stages of the work. A discussion of this matter can be based on a consideration of the following factors, for each stage of the CATP:

1. Capital investment.
2. Running costs of equipment (exclusive of salaries of operating and maintenance personnel and of personnel and parts costs for repair work).
3. Running speeds of automatic equipment in error-free production, and working speed of personnel.
4. Error frequency of equipment.
5. Time required to detect, diagnose, and repair equipment faults.
6. Difficulty of correcting errors in material that has been processed.
7. Rate of remuneration and guaranteed income of operating, supervisory, and "pencil-and-paper" personnel.
8. Productive lifetime of equipment, allowing statistically for loss of time due to breakdowns and maintenance (i.e. for factors 4 and 5).
9. Running costs which may be ascribed to maintenance and to repair of equipment (determined by factors 4 and 5 and the rate of remuneration and guaranteed income of maintenance and repair personnel).
10. Quality of product.
11. Potential work load.

The successive stages of a CATP, for which these factors must be considered, include

i. Writing and testing programs.
ii. Preparation of typographic specifications in a suitably coded form.
iii. Keyboard work for material which is to be set from manuscript.
iv. Auxiliary punched-tape and card handling work.
v. Computer processing.
vi. Operation of typesetting machines.
vii. Proofreading and other checking operations.
viii. Composition and plate make-up work.

Stage i requires primarily the efforts of programmers, but the process of testing programs and correcting errors until the programs are operational usually gives rise to small jobs of types ii, iii, iv, and vi, and may require jobs of type v that add up to a substantial total. This matter is discussed in greater detail in later sections of the present chapter, but it may be mentioned here that at times, the development or purchase of programs constitutes an item of capital investment in "software" that is quite comparable in the analysis of its consequences with capital investments in equipment.

Stage ii requires "annotators" and similar personnel, but no equipment is needed, so that only factor 7 applies.

Stage iii requires keyboard equipment and keyboard operators.

Stage iv requires auxiliary tape and card equipment, and operators for this equipment.

Stage v requires computers and computer operators.

Stage vi requires typesetting machines and personnel to supervise their mechanical operation.

Stage vii requires personnel but no equipment.

Stage viii requires personnel and some equipment.

The work load in an installation that is equipped for CATP's will depend on the quality of the product, on the costs to the "customer," on the times which he must wait for jobs to be processed, and on the demand for typeset material of the variety with which the CATP can deal. The costs and times depend directly or indirectly on the factors 1 to 9, and can be analyzed in terms of the following composite and further factors, as applied to the successive stages of the job:

12. Equipment setup time.
13. Error-free production time, which depends on factor 3.
14. Rerun and "incorporation of correction" times, which depend statistically on factors 4 and 6.
15. Test run time, prior to production runs, in novel applications.
16. Waiting time for equipment because it is in use or has broken down.
17. Waiting time for personnel.
18. Equipment time, as determined by factors 12 to 15.
19. Personnel time, as determined by factors 12 to 15. For both this and factor 18, a customer may be charged on a prorated cost basis.
20. Elapsed time, that is, the time which elapses from the occasion when a particular stage of a CATP could be started, if equipment and personnel were available, to the occasion when that stage of the CATP is finished. This is affected by factors 12 to 17.
21. Profit (or loss) that is considered acceptable in the stage of the CATP that is being analyzed.
22. The rate to be charged for the use of equipment in any stage of the process. This is determined by factors 1, 2, 8, 9, 11, and 21.

23. The profit (or loss) that is considered acceptable on personnel who deal with the stage of the CATP that is being analyzed.

24. The rate to be charged for personnel time in this stage of the process. This is determined by factors 7 and 23.

Factors 18, 19, 22, and 24 taken over all stages of a CATP determine its cost to the customer. Factor 20, that is, elapsed time, taken over all stages of a CATP determines the total elapsed time which the customer must wait for the end product after submitting a job to be processed.

Mention should be made here, too, of three, further factors:

25. Backup availability of similar equipment in other installations.

26. Compatibility of equipment with that which will be manufactured subsequently.

27. Availability of trained personnel and potential for training further personnel for operation and maintenance of equipment and for pencil-and-paper work.

These affect the stability and scope for expansion of a CATP operation, and this together with expected workload and net profit bear on the desirability of the initial investment that must be made.

Some of the factors which have been mentioned in this section are discussed further in the Sections 9.4 to 9.8 which follow. The separation of certain of the factors in particular circumstances is rather arbitrary—for example, some equipment may be maintained and even repaired by the operating personnel.

9.4 Programming Costs

The personnel and computer times that can be spent in writing, testing, and correcting programs must not be underestimated. In the application of computers to many fields, "general purpose programs" are written to deal with whole classes of problems, at the cost of relatively little incremental work per individual job. For example, the TYPRINT option allowed many different typesetting jobs to be run without the need for further programming. This principle of using parameters to specify typography in the input to a general typesetting program has been presented now to the typesetting industry by many exponents of CATP's. In practice, however, it is very often necessary to augment the initial investment in a general purpose program by continuing program maintenance and supplementary programming. Program maintenance consists of finding and correcting errors when these first come to light and making minor modifications and additions to a program as experience is gained, and as changes are made to equipment and to procedures. It is an activity that by and large is necessary in an environment which evolves and grows. Supplementary programming consists of the construction of relatively small programs which are written for individual jobs and which are added to the larger and more general programs to enable these to cope with the special circumstances of the jobs in hand. Several examples of supplementary programming are given in earlier chapters. Computer usage tends to flourish in organizations that employ personnel for program maintenance and supplementary programming even when they use some general purpose programs of considerable power. In fact, the use that is made of a general-purpose program often depends on the ease with which the program maintenance and supplementary programming can be done by the user's staff. An example is to be found in the provision of compilers by computer manufacturers. A major selling point of some computers has been the availability of compilers for them that not only work but are also designed and documented to allow maintenance, modification, and supplementing by user staff. It should be noted that groups providing general-purpose programs are often best advised to encourage and enable the user's staff to do as much as possible of the tailoring of the general purpose programs to the user's special needs. The providers of the general-purpose program can otherwise be overwhelmed by a multitude of individual requests that must be met to make their original efforts really beneficial. These comments are not meant to imply the nonexistence of specialized jobs of very high volume which really do not change in time and which would need only negligible program maintenance and supplement. The author would urge caution, however, to those who think that they have such a job in prospect.

There are various ways, of course, in which one organization can benefit from CATP's without bearing in full the cost of all the programs which are used in the jobs they run. General-purpose programs for use on a particular computer or for a particular typesetting machine may be provided by the equipment manufacturer and the costs met from their overhead. A very large organization may provide its departments with programs the costs of which are met from overhead or by special funding. Academic and similar research groups sometimes contribute usable programs to the public domain. Consulting agencies also market programs which they develop as proprietary material. With respect to this last activity, a large general-purpose program may require widespread use to recover the investment that was made to write it. The protection of proprietary programming, however, does not always accord with the dissemination of the contents of the program in an intelligible form, which as suggested above, may be necessary for a supposedly general program to receive really widespread and effective use.

9.5 Equipment Costs

If equipment can be used full time and its operating staff fully employed, the significant cost factors in the application of the equipment to a given job are the following:

The time required to run the job.
The ratio of equipment purchase price to its effective working life.
The pay rate of the operating staff.
The maintenance, materials, power, and other running costs.

The capital investment that is necessary, however, can be of paramount importance at the inception of a tentative effort of uncertain outcome. It transpires that with some forms of equipment, notably general-purpose computers, the cost per operation goes down as the total cost of the machine goes up, provided that the cost of a run can be prorated by reference to just the time that it takes. Although it may not be possible to justify a large computer for CATP's in a particular installation, it may be possible to share the time and cost of such a machine with other work. This could be done, for example, in the central computer installation of a large company, or a governmental or educational establishment, or in a service bureau. Again, it might not be possible to justify a high-speed typesetting machine for the work that is keyboarded in a particular organization, but if there was a sufficient work load of computer output to be typeset, then modifying the keyboard operations to take advantage of computer aids and of the fast typesetter could be expedient. Service bureau operations remove the burden of capital investment from the user, though he pays a price for this. Problems of communication, of transmission of material, and of scheduling delays tend to be lessened by the installation of equipment on the user's premises and under his control. Developments in computer time sharing and in data and image transmission will increase the usability of large and remote equipment. CATP's may provide one of the first areas in which "public utility" computing can be introduced in an economically beneficial way, and attempts to use this approach in CATP's could well provide practical insights that are of value to the development of public utility computing as a more general activity.

Environmental factors and financial policy sometimes preclude heavy capital investment while allowing further personnel costs and nondepreciative running costs. At times the converse holds. Individual partial cost considerations can outweigh total cost, say in determining the choice between a special-purpose code translation device and the continuing use of a general-purpose computer for code translation, if a budget does not provide for the purchase of further equipment but does allow a good margin for the use of more computer time. Effectively free or "reduced price" use can sometimes be made of unsaturated equipment whose presence is required for other work.

Many industries and individual organizations make use of a spectrum of computing equipment. There is no reason for supposing that the typesetting industry will be any different if it finds the widespread use of computers advantageous. By analogy to the computer industry, too, in which sales of small and large machines progress apace, the increased use of very high-speed typesetting machines should not preclude the increased use of equipment of conventional tape-operated size. Code compatibility would be advantageous.

Guaranteed wages require consideration separate from hourly pay rates for much the same reason as the separation of capital investment from prorated equipment running costs. There is no advantage, from the standpoint of labor costs, if personnel time per job is decreased and the work load does not increase while a guaranteed wage is maintained. Moreover, organizations that use computers in most fields of endeavor tend to require their own programming personnel, and this is also likely to be true in computer-aided typesetting. A programmer requires a full salary even if he does not have a full work load. Consulting agencies parallel the activities of service bureaus, and often are associated with these, but their programming charges are quite appreciable.

9.6 Over-all Production Speeds

In assessing the over-all production speeds, it is essential to consider the probable waiting, breakdown, and operating times for machines and staff at all successive stages of the job, when it is run correctly and also when mistakes are made that require reruns and the incorporation of corrections. Attention should be given in connection with this, not only to the probability of breakdowns and errors, but to the delay in their detection, and to the availability of personnel to deal with them.

Separating different stages of a job onto separate items of equipment sometimes enables greater individual speeds and more continuous operations. For example, typesetting machines which could be operated directly by a keyboard are often operated instead by paper tape that is prepared on separate keyboard perforators. This allows the keyboard operators to work in more congenial, and at times remote, surroundings and at higher speeds. It also enables the typesetting machines to run faster and more continuously than when they are operated directly from an attached keyboard. There are occasions, however, when it is more convenient to operate a typesetting machine directly from a keyboard than by a two-stage process. The objective of much computer work at present is to allow one-stage processing of jobs that previously required waiting for, and using, several successive machines. Difficulties that can arise in different stages of a job may magnify each others consequences. A safe system provides buffering to accommodate and contain local errors and may require rather more over-all time than the sum of estimated error-free production times for the successive stages of the job. Even more time will probably be needed if the possibility of errors is not recognized and an accident-prone operation is attempted. Times and costs can be dominated by what goes wrong, when allowance has not been made for errors, and CATP's seem to offer vast scope for things going wrong, from the author to the binder.

The turnaround time is of varying importance in different circumstances. In competitive newspaper work, minutes are vital. It makes little difference, however, to a research project that has gone on for years, if the results take a day or two longer to typeset (unless a conference deadline must be met). When programs are being tested and corrected, quick turnaround is very important, to morale. In some circumstances, it is good enough to plan for a certain average turn around time, without disastrous consequence if this occasionally is not met. In other circumstances, elaborate insurance must be taken to ensure the maintenance of operations within a narrow tolerance.

9.7 Keyboarding

The benefits of a CATP seem most obvious when the computer removes the need for keyboard work, for example in CATP's which set results of numerical calculations, or of nonnumerical processes such as mechanized algebraic transformation and language translation. In these cases, the computer output consists of material whose literal content has not been keyboarded, so there is no question that keyboarding, transcription errors, proofreading, and so forth are avoided. The programming cost may be high, and for short runs with complicated typography and make-up they may be uneconomical. The examples of such work that were run with the PC6 programs, however, incurred costs that, including the supplementary programming, were appreciably less than the estimated costs of setting type by conventional methods. Further CATP's, such as the MEDLARS project, which set keyboarded material with extensive mechanical repetition and/or rearrangement, unquestionably benefit from the computer's aid and have considerable social importance. The aspect of computer-aided typesetting, however, that seems to have engendered most widespread concern is the simplification of keyboard operations by mechanized hyphenation in straight text-matter work. Small computers are being used for such applications in several newspapers and some book plants in the United States. The economics of using a computer to speed up keyboard operations, in the absence of computer organizing or updating processes that would cut down total keyboarded text matter for a succession of publications, seem to depend on several factors that may be rather sensitive to the character and environment of the material that is set. The cost of the keyboard work for a given job, given an adequate total work load, depends primarily on the ratio of the purchase price of the keyboard equipment to its effective lifetime, the pay rates of the keyboard operators, and the time that they take to type the job. This is a partial cost to which the costs of other operations in the over-all typesetting process must be added to assess net benefits, and it can be analyzed further to a dependence on the number of key strokes to be typed for the job and the average rate at which these are typed.

The number of key strokes depends on many factors:

1. The number of keys that are available to represent characters (e.g. whether upper- and lower-case characters are typed on separate keys or by the use of a shift key, and if there is a shift key, whether this always locks).
2. The number of text characters to be typed.
3. The incidence of case shifts when case-shift keys are used.
4. The number of keys that are available to represent typography and format.
5. The coding conventions that are used to represent typography

and format with these keys.

6. The incidence of typographic changes that must be represented.

The rate of typing depends on these factors 1 to 6 and also on the following:

7. The keyboard arrangement.
8. The key stiffness.
9. The auxiliary actions and decisions that the operator must take.
10. The maximum proficiency that the operator has ever achieved with the equipment in use.
11. The operator's current adeptness, which may differ from 10.

Factors 1 and 4 to 9 determine in a sense the innate "difficulty" of the work. Too many keys to press can be confusing to an extent that outweighs the reduction in key strokes which they provide. The relative position of keys affects their convenience. These and matters such as line breaks can make the operator pause to think, and these factors all in turn may affect the innate intellectual capacity that is required, the training and the practice periods that are necessary to achieve proficiency, the likelihood of losing this proficiency if it is not maintained as a full-time occupation (perhaps by alternation with nonkeyboard work or work that uses other keyboards) or if it is completely interrupted for some period of time, the recouping period needed in the latter circumstance to regain full proficiency, the maximum keyboard speed that can be achieved with the given equipment, and the likely error rate.

The computer can affect these issues since it allows a choice in the keyboard perforator that is used, in the coding conventions that represent typographic changes, and in the number of decisions that the operator takes. For certain forms of material, the computer can reduce the number of key strokes that are used to control typography. This happens when a very standard format is used—for example, short news items in a newspaper. The initial typographic information, which in TYPRINT style input is specified by codes typed explicitly at the beginning of a text, then can be omitted by the keyboard operator and provided automatically by the program. A program that basically has a greater generality can be used in this way by association with a specialized preprocessor or initialization subroutine that sets up the appropriate typographic parameters at the beginning of each run. Alternatively, a control card or a control tape that is punched with the relevant codes could be read into the computer ahead of a text which was typed without initial typographic control information. This last measure in fact could have been adopted with the TYPRINT option. Again, the selection of one of several options could be made by typing a short reference code at the beginning of a text—rather like a macro whose expansion is implicit in the preprocessor or initialization routine—or by the provision of an appropriate control card by the computer operating staff. It is of questionable benefit, however, to remove the need for the keyboard operator to type a code in the tape if this is replaced by his need to write an instruction to the computer operator, and the need for the computer operator to select the appropriate control card. The expenditure of personnel time is still involved, and the system is made a little more accident prone by the need for a communication that may be incorrectly expressed or acted on.

Similar arguments apply to typographic controls within a text. A few keys that are not committed to text characters can deal simply with a few operations in a job of limited typographic change. The same program could deal with several classes of job which required different, but small, sets of typographic changes by allowing the same set of tape codes to be assigned to these different sets of changes, either (1) by an initial code that caused the preprocessor or initialization routine to bring the appropriate set of equivalents into play, or (2) by a control card or tape that was included in front of the text to be processed, or (3) by macro style definitions that were typed by the keyboard operator. But again, alternative (2) puts more work on the computer operator, and alternative (3) puts more work on the keyboard operator and on the annotator too. And as soon as the number of typographic controls within a text passes say half a dozen, a coding language that requires several key strokes per typographic action, or a keyboard with a considerable number of keys, becomes necessary.

These comments are not meant to decry CATP's for straight text matter but only to suggest that the argument of reduced key-stroke action should be reviewed with caution by potential users and by programmers, particularly when a diversity of jobs is contemplated and it has been demonstrated for just one or two of these in isolation. Of course, the argument of reduced key-stroke action does apply at times, and even when it does not, there are further factors that may warrant the use of a CATP.

A reduction in the frequency with which the typesetter must make decisions has been the main motive for applying the computer to the hyphenation problem. It is clear that fewer decisions are made when a perforator which produces no hard copy (such as the teletypesetter perforator) is typed continuously, than when the operator of such a machine watches the pointer, types line-break codes, and hyphenates. Some straight matter CATP's, however, use keyboard perforators, such as the LCC perforator, which do type hard copy in which lines are broken without reference to where these will occur in the typeset end product. Success with this approach would suggest that the decisions that are made to break lines in the hard copy are less demanding than the line-breaking decisions for the final typeset product. They need not be so precise or frequent, and hyphenation is not necessary. This is perhaps a matter that could receive some further analysis, and it may be affected by advantages of the perforator which do not depend on the computer's intervention.

Line-breaking and hyphenating programs are relevant to CATP's which produce typeset revisions of material that is updated or edited by the computer. Changes of text can alter the position of line breaks, so that even if the first edition was typed say on TTS with line breaks put in by the conventional method, these would be subject to alteration in the editing process. The line-break hyphenation problem, however, is not committed to a computer program even in this instance. For example, editing could produce a tape without line-break codes which was then processed by a special-purpose device, such as Linosec (see Section 8.6).

The line-break and hyphenation problem has been discussed extensively in various media already, but the author has not made any contribution to this, or to the problem of rivers, which can be cited here. He is skeptical that reading habits will be changed, that conventional typographic standards are unnecessary, or that standards of hyphenation are as stringent as sometimes claimed. An obvious solution to the problem of hyphenation and rivers is to have a program that breaks lines, hyphenates, and avoids rivers in a reasonably effective manner, with the facility for subsequent incorporation of local changes if the initial product is not acceptable. Economical methods of getting first proof would help in this regard. Practical considerations such as available programming effort, lead times, and computer storage determine the breakpoint which must be adopted between programming for first pass typographic quality and relying on the possibility of second pass amendation. A system that does not permit local changes to be made easily is unrealistic for a majority of applications. If a system does allow changes to be made easily, this facility would enable the distribution of material between lines to be altered in a second pass when necessary.

Moreover, the author suspects that the most sophisticated river-deleting and breaking program will not vie economically, in the foreseeable future, with the power of the human eye and intellect to detect a multiline situation in which moving words will just move rivers, but changing a word without altering literary content or value is remedial.

It is possible that CATP's may sometimes lessen the frequency with-which keyboard operators must make decisions when setting typographically complex work. Many of the marginal comments which copy editors make on manuscripts, and which require interpretation and action by the keyboard operator on the special keys of complicated keyboards, can be expressed by control codes, such as the TYPRINT codes, which can be typed on simpler keyboards. For practical applications, more concise conventions than those of TYPRINT are needed, but the principle can be the same. Annotated manuscripts could be typed without the operator pausing to interpret instructions. Text characters and control code characters would be typed by the identical visual-muscular processes. This could be sufficiently advantageous to outweigh a substantial increase in the number of key strokes. Potential CATP's of this sort, however, require careful analysis. The density of material on a manuscript may make it difficult to write the control codes between the text points at which they apply, and the departure from continuous left-to-right scanning that would be needed to observe marginal inserts could detract from the usefulness of this approach. Again, the keyboard operator makes some decisions that could not be described by control codes of the TYPRINT variety in a practicable fashion. For example, tables that are laid out by inspection and judgment would require special programs if decision making at the keyboard was to be significantly reduced. Particularly where questions of composition are involved, the human combination of eye, intellect, and aesthetic experience are all important, and the intervention of the computer can become a hindrance if this is not fully recognized.

A final issue that may be discussed here in relation to keyboard practice concerns the use of standardized keyboard equipment within a typesetting installation. For many years, typesetters had to deal with just manual linecasting and monotype and teletypesetter keyboards. The number of different models of typesetting machine and associated keyboard perforators has increased considerably in the last few years. The advantages of CATP's that allow one keyboard perforator to punch tape for different forms of typesetting machine were mentioned in Section 9.3. From the standpoint of keyboard personnel, these CATP's avoid need for retraining and the possible loss of proficiency on one machine when another is to be used. This in turn affects job mobility, continuity of operation when one model of machine is replaced by another, and versatility when one installation deals with several models of typesetting machine concurrently. The fact that the computer offers this flexibility, however, should not obscure the fact that special-purpose recoding units at times can be more economical than a computer, for simple code translation from a single keyboard perforator to several typesetting machines.

9.8 Typesetting Machines

The possible effect of computers on the design, cost, and speed of typesetting equipment may be mentioned briefly. The codes that operate hot metal typesetting machines are punched in tape by fairly simple mechanisms which are activated by keyboards. The codes which operate the character projection devices in filmsetting and electron-beam typesetting machines, however, must be formed from the primary effects of key depression in much more complicated ways. Relay and electronic circuits are interposed between the keyboard and the character projection device to deal with these. Thus, the Photon 200 interposes what is in effect a special-purpose relay computer between the keyboard and the photo unit. In other filmsetting machines, electronic circuits are interposed between the keyboard and the tape perforator, and between the tape reader and the photo unit. Arithmetic operations are needed at times to compute escapements. The optical principle on which the Photon ZIP (the unit called the GRACE machine at MEDLARS—see Section 8.7) is based requires the sorting of the characters in each line into a different sequence from that in which they are keyboarded or produced in the initial representation of computer-generated results which are to be set in type.

It is possible to use either special-purpose devices or general-purpose computers to perform the arithmetical and logical processes which convert sequences of character codes into the sequences of codes which operate the character projection units of photographic and electron-beam typesetting machines. The consequences of using a computer to perform the operations that otherwise are performed by circuitry in a keyboard perforator unit have been discussed already. A computer may also perform operations that otherwise are performed by circuits in a tape operated typesetting unit. This may allow the elimination of circuits in the typesetting unit, with a lowering of cost. The special circuits that convert sequences of character codes to the codes which control the character projection device delay the latter's actions in some tape-operated filmsetting machines. The filmsetting process then can proceed more rapidly if the character projection circuit is operated by tape in which a computer has punched the codes that control its actions directly. The reduction in the cost of the typesetting machine and any increase that is possible in its speed of operation when a computer is used to deal with processes that are otherwise performed by special circuits in the typesetting equipment must be compared with the attendant computer costs to determine whether or not they are advantageous. If the sequences of character codes are produced in the first place as the result of numerical calculations or of non-numerical processes which are to be typeset, then further processing by the computer to give character projector codes merits strong consideration but is not inevitably beneficial. There are advantages to a single typesetting unit which can be operated both by computer-generated tapes and also by tapes that are punched on a keyboard unit without requiring the computer's intervention. There may be circumstances, too, in which the production of the character projection codes is best effected by a unit that controls the character projector on-line, and in which it is more economical to use a small general-purpose computer as the controlling unit than to use special-purpose circuitry for this purpose. Situations would then arise in which one general-purpose computer produced results to be set in type and wrote these as a sequence of character codes on tape which then served as input to another general-purpose computer that performed the further conversion into character projector codes and operated the projector directly.

10. Administrative and Training Problems

10.1 Introduction

The exploration and introduction of CATP's require the provision of suitable training programs for the personnel who will deal with the several aspects of the work. It also raises several matters of an administrative nature. The new skills and changes in skill that will be needed are listed in Section 10.2. A way of presenting CATP's that has been found useful and which might be adopted as the prototype of a course of instruction is analyzed in Section 10.3. Some questions concerning the effect of CATP's on the employment of keyboard operators are considered in Section 10.4. The tactics of programming are discussed in Section 10.5. Section 10.6 contains some comments on the interaction of technology and academic research.

10.2 New Skills and Training

Computer-aided typesetting processes will require personnel to deal with several tasks, of which some have not arisen previously in any context, some are modifications of tasks that already exist in the typesetting industry, and some have existed in other industries but not in the typesetting field hitherto. CATP's moreover may reduce the extent of some tasks that are performed at present and in some cases eliminate these entirely. The demand for new and changed skills and the diminution in demand for certain existing skills raise several problems that are of general concern to the printing and publishing industry.

CATP's will require personnel to fulfill the following tasks:

1. Perform the over-all system design of CATP's.
2. Specify, design, write, test, maintain, and modify programs.
3. Prepare typographic control specifications, for example by annotating manuscripts.
4. Keyboard text material, programs, and typographic control data.
5. Operate auxiliary card and tape-handling equipment.
6. Operate computers.
7. Supervise tape-operated typesetting machines.
8. Maintain and repair keyboard, and auxiliary card and tape-handling equipment, computers, and typesetting machines.

CATP's may reduce the need for personnel to keyboard text material, and change the keyboards and procedures that are used in keyboard work, relative to that which is done at present. Since CATP's may increase the extent to which filmsetting methods are used for material which at present is set by hot metal, there may be local reductions in the demand for plate-making personnel.

Systems design will require personnel who combine insight and understanding of the over-all operations of the printing and publishing industry with an appreciation of the computer's potential and of the practical issues that are involved in using a computer. It is to be assumed that such personnel will obtain their expertise in general by adding a knowledge of computing to a relatively long-term experience of the printing and publishing industry. It would be appropriate and quite feasible, too, for some personnel who are employed at present as typesetters and compositors to undertake part of the work of categories 1 and 2 in the foregoing list, to share the work of category 3 with editorial and typographic design personnel, to deal with most of the work of category 4, to deal with the work of categories 5 to 7 in CATP's, and to deal with the work of category 8 except where it requires knowledge of electronics.

10.3 A Prototype Syllabus

It is of considerable importance that personnel in all branches of the printing and publishing industry obtain an understanding of the possibilities of CATP's. Mention may therefore be made here of the form of presentation of the basic concepts of CATP's, which the author has found to be of use in conveying these concepts to personnel in the industry who have had little occasion hitherto to interact with computers (in the form of unpublished notes by the author, entitled "Introductory Ideas for Computer Typesetting"). Emphasis is placed on technological continuity. The presentation is developed in the spirit that there is no sharp boundary between the sorts of things that are being done at present without the aid of computers and the sorts of things that will be done in the future with their aid. "There will be differences of size, and of emphasis and detail. Typesetting with and without computers should not be viewed however as in two isolated islands of endeavor, without common traffic or communication. The potentialities of the computer can be fulfilled to the benefit of all along gentle roads that do join smoothly with the paths of present practice, even though these new roads may continue far beyond the horizons which are now in sight. Computers can be arranged along a road in which some rather small and better-known machines come first." As a preliminary to comments on computers and to provide an introduction to the basic concepts of CATP's, the presentation then goes on to discuss several machines:

1. The cash register, which prints items as they are keyboarded and totals when an appropriate key is pressed.
2. The National accounting machine, which is used extensively in banks and in other commercial organizations, and which was widely used to construct mathematical tables before the advent of computers.
3. The TTS perforator and tape-operated linecasting machines.
4. The pianola—a modified piano which is operated mechanically by perforated paper scrolls.
5. The Flexowriter.
6. The Listomatic machine for catalog production.
7. The Hollerith card punch, tabulator, sorter, reproducer, and collator.

When an idea is explained by reference to the use of some piece of equipment, many people have difficulty in assimilating the idea if they feel that they cannot understand the internal workings of the equipment in question. Many people in the printing-publishing industry who think that the internal workings of electronic computers are outside their comprehension could nonetheless be given confidence in their ability to understand the internal workings of all the machines which have just been listed. Since the basic ideas of CATP's can be presented by reference to these simpler machines, it is of some advantage therefore to do so, in order that the ideas be assimilated as widely as possible and to stress the element of continuity which was mentioned in the preceding paragraph.

The cash register illustrates the mechanical recording of information which has not been keyboarded explicitly (i.e. totals) but which has been formed instead by the machine from other information which was keyboarded (i.e. individual charges).

The National machine can be used to illustrate the mechanical generation of typed output that is much more voluminous than the information which is first keyboarded. A very simple example which requires the merest knowledge of arithmetic is the automatic production of a table of multiples of a number that is set on the keyboard, by repeated addition of this number into the accumulator and automatic recording of the successive totals. An example which is only a little more complicated and which has been well received in presentations to senior keyboard operators is the construction of a table of squares from their second differences.

The pianola illustrates the principle of controlling the successive operations of a piece of equipment by the positions, across a roll of paper, of perforations that are encountered successively along the length of the roll. The mention of the pianola is helpful in introducing the idea of tape operation to older typesetters and compositors who are not familiar with TTS or Monotype practice. Mentioning the pianola to personnel who are already familiar with tape- and spool-operated typesetting equipment helps to emphasize the diversity of machines which can be controlled by perforations in a strip of material, such as paper tape, that is moved through a sensing mechanism.

The TTS perforator and the tape-operated linecasting machine illustrate the basic use of perforated tape in typesetting. The distinction between codes that select characters to be set in type and codes that cause other machine operations, such as line-end, can be explained very simply by reference to this equipment.

The usage of the Listomatic machine illustrates the principle of maintaining a large number of independent items of keyboarded text in a way that allows any necessary selection to be put together in any necessary order in a list that is published without the need for further keyboard work. The items are typed (or printed) on separate pieces of card that are filed. The updating, selection, and arrangement of items for publication entails the updating, selection, and arrangement of cards in a file that is then processed by the Listomatic machine. This idea can be presented first in terms of manual control of the card files. It can be extended later, after the discussion of Hollerith machines, to the mechanical control of Hollerith cards on which items of text material are printed or typed. Identifying codes that are punched in these Hollerith cards can be used to select and arrange the items to be included in individual publications. The idea of organizing cards mechanically for the Listomatic process can be used still later to lead into the idea of using a computer to process items of a list that have been punched in Hollerith cards, to update, sort, and perform other organization processes as a preliminary to mechanized typesetting.

A discussion of the Flexowriter can be used to introduce several further ideas. A tape that has been keyboarded once can be used repeatedly to type duplicate cards for an item that is to appear under several headings in an index that is prepared by the Listomatic process. This use of the Flexowriter thus provides an example of a mechanized process of the variety that is called "repetition" in the discussion of organization processes in Section 9.2. Producing an updated tape on a Flexowriter by copying appropriate portions of an old tape mechanically and typing manually the portions that have changed can be presented as a prototype of mechanized editing. It is conceptually simple, and computer editing can be presented subsequently as a process that is qualitatively similar to Flexowriter editing but rather more powerful in its details. For example, computer editing provides the facility to readjust line breaks in order to accommodate deletions and insertions, and it enables the keyboard operator to type corrections continuously, without having to wait between corrections while tape is copied mechanically.

The fact that the Listomatic and the Flexowriter can both be used to produce printed or typed material in which portions that have not changed since an earlier edition are not re-keyboarded may be stressed in a presentation to personnel who think that CATP's innovate the avoidance of re-keyboarding material that is unchanged between editions. Such personnel may be reminded of the fact that although computers may decrease the amount of re-keyboarding between successive editions of a publication below the amount that is necessary now, it is not the computer, or the Flexowriter, or the Listomatic machine that has created a situation in which new editions are not re-keyboarded completely. It has been the practice for some considerable time to save standing type for new editions.

Punched-card machines complete the range of equipment which the author has found it expedient to describe as a preliminary to CATP's. The card punch provides a variant on tape and spool perforators as a means of keyboarding text in machine readable form. The tabulator provides an easily understood mechanism for "printing" the contents of a deck of cards. The sorter, reproducer, and collator provide easily understood mechanisms for performing the "organization processes" of Section 9.2 on material that has been punched in decks of Hollerith cards. A simple example that was used in several presentations deals with the membership file of a club that contains a card for each member. Each card carried the member's name, address and date of joining. Membership lists are maintained in alphabetic order of member's name, in alphabetic order of street address, by chronological order of date of joining (for determination of seniority), and by month of joining (for annual subscription renewal). Alternative ways of maintaining these four lists illustrate the use of the card punch, sorter, reproducer, collator, and tabulator.

A two-day presentation of CATP's which has been found quite effective consists of (1) a discussion of several simple machines, as outlined in the preceding paragraphs of this section, (2) a demonstration of a Flexowriter and of Hollerith punched-card machines, (3) a short visit to a computer installation, (4) an account of an interspersed coding system such as TYPRINT, (5) a discussion of the need to set computer-generated material illustrated by examples such as mathematical tables and the *Index Medicus*, and (6) a discussion of general questions of the practicality of CATP's.

It seems too that a course of, say, three weeks could provide suitable personnel with sufficient training to enable them subsequently to participate with understanding in experimental studies of CATP's and by such participation to learn more about computer usage and assume an increasing responsibility for the tasks that are enumerated at the beginning of this section. Personnel for such courses and subsequent work could be selected from keyboard operators, compositors, and supervisory staff in the industry. The course could consist in essence of an expansion of the two-day presentation that is described in the preceding paragraph. Sufficient time would be given to operating the Flexowriter, card punch, and card-handling machines and to simple control panel wiring principles to enable the class to perform simple tasks with these machines, in accordance with specifications that they were given. The course could be given to small classes, and the students selected in part from mature staff in companies that wished to experiment with CATP's and in part from trainees in teaching establishments of the printing-publishing industry. It could be followed by experimental studies of CATP's in which each member of the course would work under supervision of research staff who had been involved in CATP's for some time. The experimental CATP's could be based on existing programs so that the personnel new to computing could get their initial experience by making small additions and changes to existing material, rather than by having to undertake major tasks from the outset.

10.4 Employment Effects of Changed Keyboard Skills

The discussion of training and skills for CATP's has not dealt so far with the aspect of the problem that seems to attract the most widespread concern, namely the employment implications of changed keyboard practice. The author has emphasized the other aspects of the problem quite deliberately because he believes that there is much more to CATP's than an avoidance of hyphenation by keyboard personnel. This issue is, however, of considerable importance, and as a result of the simplified keyboard operations which it will allow in the avoidance of hyphenation and in other respects, the computer may affect several aspects of the employment of keyboard operators:

1. The reduction in operator hours needed to deal with the present volume of keyboard work.
2. The increased volume of keyboard work that can result from the net lowering of costs per job to the customer.
3. The ability of present keyboard operators to adapt to the new keyboard procedures.
4. The disadvantages to present keyboard operators of adapting to the new keyboard procedures.
5. The possibility of the keyboard work being done by personnel who are not typesetters with traditional training.

Increased typing speeds may be reflected in either increased hourly pay rates, increased profits per job, or lower costs to the customer, or in some compromise of these. Income to the operators depends both on hourly pay rates and on working hours, so that in principle, production may be maintained and perhaps increased, with an increase in leisure, if there is a sufficient increase in hourly rate.

In some sections of the industry, it is possible that reducing the costs to the customer will not lead to an appreciable increase in the volume of type, so that the demand for keyboard operators would fall. It is to be hoped that where this was the case, staff reductions would be made by decreased recruiting and by the transfer of personnel to other activities which they would accept readily. It is to be hoped, however, that faster typing would be reflected more often by an increase in demand that resulted from the lower cost per job, and which allowed an increase in the level of employment, incomes, leisure, and profits, or at least in increase of some of these without any drop in the others. There certainly seems to be a need for detailed market analyses to determine the extent to which the volume of work would be increased by lower costs for the various forms of published matter. The author is led also to inquire if such benefits might be gained, in particular instances, by increases in the productivity of keyboard operators which could be made without recourse to computers.

As regards the ability of keyboard operators to adapt to new keyboard practice, the sort of change which may be involved would be no more extreme than say the change from manual linecasting machines to TTS perforators or to Lumitype (Photon) 540. The industry presumably has abundant data on the ability of operators to retrain for these changes and the retraining periods that are necessary. Since CATP's may remove the need, say, to hyphenate, the operator who has been breaking lines and words for years will derive no advantage from this accomplishment over a novice who has not hyphenated, when both are training for a procedure which does not require keyboard hyphenation. Experience of hyphenation, however, should not make the former operator any less able than the novice to learn the new procedures. Simplification of a process may remove the advantage which its practitioners previously possessed over other personnel, but it should not put the previous practitioner at an active disadvantage.

Retraining can of course carry disadvantages, for example when it means the loss of a skill that is required again because the new procedure proves impracticable or because the operator transfers to employment which still uses the old procedure. Retraining gives rise to many problems of organizational detail, for example during a period of transition in a large typesetting shop, when personnel who have been retrained for a new procedure which has been introduced only partially, have seniority privileges for overtime work with the older procedures. The sorts of question that can arise from keyboard retraining for CATP's are much the same as those which arise from retraining schemes throughout industry, and which have been handled in a satisfactory manner on many occasions and in an unsatisfactory way at other times. CATP's can suffer, as can any industrial innovation, from the reaction to a situation in which personnel have retrained for a process that has been introduced too hastily and which has to be abandoned because it is impracticable. Computers have simplified operations in many industrial activities, but the history of computing is paved with applications that sought to simplify and which succeeded only in making operations impossibly complicated when anything went wrong.

In connection with the retraining problem, mention should be made here of the use of the computer to achieve compatibility between a single keyboard perforator and diverse typesetting machines, as discussed in greater detail in Section 9.7.

Within the range of employment questions that the author has discussed with keyboard operators, the one that undoubtedly causes the most concern is the possibility that type may be set as a result of the keyboard work of personnel who are not traditionally trained typesetters. This possibility raises the obvious prospect of keyboard work being done by personnel who are prepared to work for lower pay rates than the present keyboard operators. It is not clear to the author, however, that the intervention of the computer brings keyboard work within the competence of personnel whose intellect, literacy, manual dexterity, physical stamina, or sense of responsibility precluded their doing the keyboard work in noncomputer-aided typesetting. In particular, the author is not convinced that the minimum standards of intelligence and literacy of personnel admitted to the industry at present is somehow higher than that required to keyboard text matter without reference to hyphenation rules. It seems much more likely to the author that the prior exclusion from typesetting keyboard work of personnel who may now keyboard for CATP's was based on legalistic restrictions rather than on intellectual or physical limitations. It seems, moreover, that certain societies impose restrictions, such as the exclusion of female personnel, that result from emotional as well as economic factors, even though economics are invoked as the sole rationalization, at times in rather odd ways. The author has heard the view expressed quite seriously by a senior trade union official that women cannot operate keyboards as well as men because women are less intelligent. Computer-aided typesetting may alter the composition of keyboard staffs, but it seems that this will result not from a lowering of the minimum intellectual prerequisites but from a breakdown of artificial restraints on recruiting and training that are more difficult to enforce in the environs of the computer.

The philosophy of remuneration is not a topic on which the author may presume to digress, so he will not pursue the question of whether innate ability or family connections should determine minimum pay rates for keyboard personnel, or whether the treatment of keyboard personnel in this regard should be affected by what labor spokesmen consider the just answer to the corresponding question for management personnel. What does seem apparent, however, is that the economic survival of an industry may be jeopardized by artificiality in any quarter of its operation. Many factors are tending to break down artificiality in the printing-publishing industry. The computer is but one. Another that may be mentioned here is the vast potential of the newly industrializing countries to set type. Keyboard personnel in the older industrialized countries are a little perturbed by the relatively low wages and the lack of procedural constraints in the younger countries. It should be noted that these latter countries have a further advantage. The scholastic level of the personnel who are recruited as keyboard operators is very much higher in some rapidly developing countries than in the older industrial societies. Advances in technology and the effective introduction of automation may very well be helped by the presence on the shop floor of personnel with the innate abilities and drives that led to a high scholastic level at school. It will be ironic if the rapidly developing countries start taking the initiative in automation because they are in a sociologically advantageous position to do so.

Although personal economic considerations are presented as the major issue in discussions of the employment effects of changed procedure, other factors are important. Just as it is possible to discern an enthusiasm at every level for innovation in a society that is imbued with a spirit of pioneering, so it is possible to find an antipathy to change in other societies, which often acts against the economic interests which are adduced as its rationalization. The roots of this attitude run deep. The author has found it interesting to try conveying the experience of participation in the development of something new to groups of senior keyboard personnel in England. He was left with the impression that these people's attitudes were conditioned by a feeling of social incompatibility with participation in such activities. Despite the fact that, in principle, educational opportunities are available in England to children of every economic background, there are very strong social factors that limit the acceptance of these in a craft or artisan environment, and when they are accepted in such an environment, a fairly strong attempt is usually made to break away from an employment that is associated with it. Interrelated factors seem to be involved in the attitudes to transition from the shop floor to management, which happens only rarely in England and when it does, is regarded almost as some form of betrayal of principles rather than as a progression. England may be particularly unfortunate in this respect. The accent of speech in England carries many of the barriers to change, passed on from generation to generation, which sociologists ascribe to the color of skin in the United States. The situation in England, moreover, may not be helped completely by the present emphasis on "technology." As a reaction to the traditional social opprobrium of technological endeavor, efforts are now being made to establish "technologist" as a title of respect. According to some recent surveys a substantial proportion of technology students feel antipathy toward shop floor employees,[33] and it will be most unfortunate if the latters' attitudes toward automation become linked to some reciprocal resentment that strove to exploit the new technologists' lack of experience of the industries which they are joining, and about which they do have much to learn.

10.5 Programming Tactics

The specification, design, coding, maintenance, and extension of computer programs is of paramount importance in any area of computer application. The technicalities of computer programming can be learned in a matter of days, without knowledge of mathematics beyond the high school level. Novices who have just acquired this rudimentary knowledge can work effectively and constructively under adequate supervision. Very often, however, a novice who has just learned to program underestimates the complexity of a job in which he wants to use a computer, underestimates the work of writing a computer program, is ignorant of the existence of extensive programming practicalities that are not taught in programming courses and are really learned only from protracted personal experience, is highly enthusiastic and optimistic about a program he decides to write, and strongly opposes any attempts to guide or supervise his work. Quite often a situation arises in which a novice in this frame of mind is indulged by a manager who has a very casual knowledge and who is overawed by the novice's enthusiasm and jargon. The practicalities of programming, like the costs, are quite considerable, however, and the fact that these exist must be recognized for the computer to be introduced effectively to a new area of industry.

A program that is of practical value usually goes through several stages of revision. The full details of a procedure that is performed by personnel exercising human intelligence may not be known to the programmer no matter how conscientiously he has analyzed the problem. Some details are brought to light only by writing and trying to use programs that are based on an initial analysis of a problem. Even when a program has been written that enables the computer to perform a process that now is carried out by hand, further changes and extensions may be needed in the program to accommodate beneficial changes in procedure that are made possible by the capabilities of the computer, and changes in the purpose of the job that result from using the computer in jobs that must precede and follow it. Developing a program thus contains a very large element of delving into the unknown, and programs therefore should be written so that they can be modified easily to accommodate unforeseen situations.

The general tactic of fragmentation (or modular approach) by means of subprograms (see Sections 7.3, 7.4, and 7.7) is used very often. Sometimes a variant subsidiary process is carried out in only two or three different ways for a considerable number of different jobs. It then may be expedient to store all the alternative variant subprograms in the computer, together with the invariant subprograms, and to select the appropriate combination of subprograms, and hence subsidiary procedures, by specifying options in the input data. Sometimes minor variations are accommodated by different portions of a single subprogram which is constructed so that at any time the computer uses only the portions appropriate to the job being run. Sometimes it is more expedient for the operator to select the cards or tapes on which appropriate variant subprograms are punched, from a file in which all the variants are stored and labeled, and to provide just the selected subprograms to the computer. Disc files attached to computers allow similar processes to be performed mechanically.

Determining the boundaries of a class of problems for which a single program or system of invariant and variant subprograms should be designed can be of considerable practical importance. The division of the entire class of problems into smaller and smaller subclasses that individually can be handled by more and more uniform processes, and the content of the subsidiary processes for which separate subprograms should be written, are important too. So also is the selection of the variant processes that are programmed first, so that some entire jobs can be run as soon as possible to start getting knowledge of the practicalities that can be learned only by trial and error, when using the computer in the new application. In large measure, the fragmentation of procedures and of programs must accommodate the programmer's uncertainties of what will have to be changed, and it must somehow separate the areas of uncertainty so that if one subsidiary process is misconceived, the programming revision can be made independently, rather than by having to make numerous changes throughout a vast interwoven structure.

As just stated, the selection of a range of problems to be treated by a single program or system of programs is of considerable practical importance. On the one hand it can happen that by aiming at too narrow a range of problems, a program may be written whose scope cannot easily be widened, whereas a program that would deal with a wider range of problems could have been written with little more effort, if the wider range of problems had been reviewed at the outset. On the other hand, it can happen that by attempting to deal with too wide a range of problems by a single program, vast effort has to be expended before any results are obtained, and then it is found from operating experience that some practical consideration that was not, or could not have been, anticipated makes the program useless.

Striking a balance between these two hazards is the challenge to the programmer's ability to appraise a field of application. In the author's view, even if attempts are made to anticipate the requirements of a wide range of problems before any programs have been written, these attempts will merely cover details of a selection of individual problems, that do not bring to light all the details of other problems of the sorts which the programmer hopes they represent. For this reason, attempting to cover a wide field at the outset may still produce a program that is rather inflexible, at much greater cost than if limited objectives had been set, as the first stage of a process of exploration and evolution.

This dynamic aspect of program development should be kept in mind at all times, for the computer to be used to full advantage. Programs should be organized as systems of distinct subroutines whose specifications and interrelationships are simple and clearly understood. Programs should be designed for intelligibility. In the author's view, this consideration transcends "tight" programming that saves a few storage locations or a few microseconds by exploiting the pathology of the computer in a way which only the person who writes the program can understand. If the job is so close to the computer's capacity or to allowable running times that these measures seem necessary, the chances are that some unforeseen circumstance will require the use of further storage locations or microseconds and that the approach which is being adopted may prove impracticable and accident prone at the least. Moreover, an intelligible program is much more amenable to improvements which are suggested by operating experience and which may allow more far-reaching improvements than the gains that usually result from tortuousness. Compiler languages such as FORTRAN, supplemented by a few subroutines in a symbolic language (e.g. FAP), provide an excellent basis for intelligible programming.

Programs should be well documented, and documentation of one program should be completed before the next program is started. Programmers often are negligent in this respect. A program that can be understood, modified, and extended, and used as a prototype by personnel other than those who wrote it is a much more substantial contribution to technological progress than the highly personal program that may allow its author to exercise a temporary tyranny over his employer, but which may also turn into an acute embarrassment for the programmer. The writer of a highly personal program is likely to find himself competing against impossible odds when he tries to justify his past efforts by ensuring the adoption of his program in a variety of user situations. He may then have to try to make separate modifications to deal with each of these situations, working single handed, while an intelligible program written elsewhere is being adapted to these individual situations by the respective user personnel working in parallel.

There are several features of program design that tend to reduce the troubles that arise in the program's use. Consistency tests on input data, on parameters that a subroutine receives from a calling program, and at other key points in the operation of a program reduce the likelihood of erroneous behavior continuing undetected for long. Meaningful diagnostics should be printed when errors are detected. Values of important parameters should be recorded at key points in the program's operation, perhaps under sense switch control, to facilitate the subsequent analysis of sources of error when the program's behavior is faulty. Error bypassing and restart procedures should be made as convenient as possible.

The need to vary the details of programs and operating procedures to deal most effectively with different job situations has been mentioned repeatedly. There are many ways in which these variations can be made, and a few of these are now listed in roughly increasing order of difficulty for the user of the program:

1. *Console switch setting.* An option can be selected by the setting of a switch on the computer console. Each of the options that can be selected in this way is dealt with by part of the program that is in store.
2. *Control card specification.* An option can be selected by the coded or verbal specifications that are punched in a control card which is read before the rest of the input. The card may be punched by the user or selected from a supply of option cards that relate to the various alternatives. Each of the options that can be selected in this way is dealt with by part of the program that is in store. A control deck that selects several options may consist of several cards.

3. *Variant subroutine selection.* An option can be selected by the inclusion in the program of an appropriate version of one or more subroutines that have been written to deal with different situations. The selection may be manual from a card file or it may be mechanized by the use of an appropriate control deck to select the appropriate subroutines from a magnetic tape or disc file on which the alternatives are written.
4. *Parameter alteration.* One or more of the constants in a program can be changed by alteration of the source program and recompilation.
5. *Table alteration.* A table of constants is changed. This can be regarded as an extreme case of 4. Depending on the circumstances, it may involve recompilation, or it may be possible to treat the table as data. A special form of table alteration is the syntax alteration which is discussed in Chapter 7.
6. *Addition.* Pre- and postprocessors can be added.
7. *Reprogramming.* Individual subroutines in an existing program can be reprogrammed.

10.6 Academic Facilities for Technological Research

When the C.C.L. typesetting project started, the author was concerned primarily with activities in the realm of nonapplied research, principally molecular quantum theory. A case can be made that this has technological implications, but in the sort of categorization that is quite usual, quantum chemistry has no difficulty qualifying as basic research, while CATP's are unquestionably technological. In the evolution of some educational systems, science has been, and still is, at a disadvantage and seeks academic acceptance by dissociation from "useful" matters except when filing grant applications. For such environments, it is easy to make quantum chemistry look so useless that it is quite free from the stigma of crass technology, but a rationalization of the study of CATP's by prime concern for the psychology of typographic aesthetics might seem a little specious. The pendulum swings, however, and technology has become the rage in certain quarters, and more centers of technological teaching and research are now projected.

The author is highly appreciative that the environment in which he worked allowed him to move back and forth between research projects of pure and technological varieties. Even though a majority of workers may confine their efforts either to pure or to technological projects, it does seem important for personnel who wish to change emphasis to be able to do so without a prior change of academic environment, and that where personnel wish to conduct parallel researches of different levels of technological applicability that they be allowed to do so, however their academic institution is classified. Technological problems that come to the attention of a research scientist who is concerned with nonapplied studies can lead him to think of basic topics for further investigation. Progress with research which was undertaken without reference to technological matters and which comes to the attention of personnel who are concerned with technology may prompt further actions on their part. This cross-fertilization of ideas and stimulus to further endeavor is helped by the mixing of research in, and teaching of, ostensibly nontechnological and overtly technological characters within a single academic institution which does not look on either as slightly out of place or inimical to its practitioner. These comments are directed toward academic institutions. While they have a bearing on research work in industrial laboratories, an employer's commercial interest provides constraints to the domain in which employees activities may range and the commitments that they must fulfill before altering course. While research work of a technological nature will occupy a major part of industrial scientific efforts, the author believes that the universities should provide a habitat for such work as well. The university can allow a freedom from certain constraints that are more or less inevitable in industry, and this can benefit the technological research. The presence of the technological research can enrich the other activities of the university. And the freedom to pursue technological research will avoid artificial boundaries to thought that would otherwise curb and frustrate the intellectual wanderlust of individual academicians.

Research work that purports to have a contribution to technology as its objective must take adequate note of realistic technological requirements. It is all too easy for an academician who has developed something which he thinks can help the progress of an industry to then proceed with further studies of "practical applications" that really are irrelevant to practical considerations, if he does not have a sufficient grasp of the problems of the industry concerned. It is of the utmost importance that proper care be exercised in the selection of practical applications to test experimental systems. in a significant fashion. It is essential to experiment with real jobs of increasing complexity to get sound insight into the real needs of industry. The difficulty is to find the industrial experts who will spend time explaining a problem and collaborating in a pilot study which can be unsuccessful without prejudice to the cause. Often, it can happen that people in industry who express willingness to collaborate with an academic project really want specific jobs run for a profit and take advantage of equipment and personnel in the university for which they need make no payment. These the universities must avoid. Technological developments, however, must fit budgets and deadlines, and the only way to find when this point has really been reached is by practical tests. It is these that require the careful selection.

The choice of a path through the possible tests and tasks of increasing difficulty is a complex matter that depends on the people and organizations with the problems as well as on the problems themselves. This choice of path and selection of collaborators can be as much a part of academic technological research endeavors as the solution of a mathematical problem or the design of a computer program. It is an element of the technological research activity which it should be the researcher's prerogative, as a delver into the unknown, to control, subject to success with his project as measured by the results that are obtained over a reasonable period of time. Of course, the research worker must be careful that knowledge which derives from work that is done with public money is publicly available.

There is a particularly pernicious form of outside influence which at times besets work of technological potential, and with which the academician must avoid involvement. Every new or impending technology has its would be entrepreneurs, who possess a propensity for learning the existence of research projects. They claim important contacts in major companies, and hold out prospects of consulting fees, research grants, technical assistance, and the satisfaction to the research worker of seeing his ideas receive widespread use and recognition. They call long distance and fly in from far-off points at little notice, for reasons of great urgency. They quote technical clichés from popular magazines. They have a capacity for claiming associations and at times extracting vague letters that imply some official interest in their schemes. They can bring suspicion on an entire field of endeavor by using whatever material they pick up to try getting contracts or backing for unsound technical ventures.

It is essential for the academic research worker to remain clearly distinct from any such enterprise of dubious technical caliber. The university must not be associated in the eyes of industry with exaggerated claims or premature attempts at implementation. Most organizations of substance include innovators and skeptics. When the innovator subscribes to some overselling and there is a fiasco, the skeptics can block for a considerable time the acceptance of sound new ideas and the cooperation that is essential to significant progress.

Coordination is an issue that crops up quite often in technological research. Duplicate expenditures on efforts that individually would suffice seems wasteful. If total funds are short, their distribution between parallel efforts may lead to inadequate support for all. Incompatible and wrongly oriented systems may result from inadequate interest in, or facilities for, communication. The "unified approach" of course is a glorious ideal. It is not clear, however, that this is at all meaningful in the development of a new subject, whose complexities and potentialities make it impossible to anticipate the full consequence of different methods of attack. Committees can discuss interminably without answering questions that require the development of working systems, and their trial under operating conditions, for resolution. Individual workers should be given the mandate to develop their own ideas in their own way if there is evidence of research competence and responsibility from past proved performance or other adequate criteria. An intellectually live technology is more likely to be developed and an industry benefited by the parallel efforts of several groups working with the courage of their convictions and enthusiasm for their own endeavors, than within the constraints of edicts issued by a committee of ignorance.

In the last resort, the technological progress of a society is dependent on its administrative beliefs, and in considering transatlantic differences in these matters, the juxtaposition of two items in a recent issue of *The Times* (of London) seems worth a little comment. An article that dealt, in fact, with book supplies for new nations stated, in discussing a research project which it thought should be supported, "America has one great advantage . . . funds are always available." On the same page, the editorial said of the United States, "Their political and administrative structure is still somewhat inchoate." In comparing research activities in the United States and England, it has appeared to the author that America's great advantage is its administrative structure and that this enables funds to be used in U.S. universities to good advantage. *The Times* editorial explained the difference between British and American administration by "Britain tends to take a high level of public honesty for granted . . . Americans make no such assumption." Whether or not Britain is really so optimistic, and if it is, whether the optimism is justified, are matters that cannot be discussed here. There is no doubt, however, that Britain places faith in an oligarchy which is not constrained by checks and balances of the sort which exist in the United States. Quite apart from any question of combating dishonesty, it can be argued that these checks and balances are needed at every level to ensure effective communication, discussion, and innovation in technology and administration, and the fair noting of excusable mistakes. Honest and competent men are fallible and have prejudices. Parallel lines of communcation to decision-making bodies help to secure adequate consideration of proposed changes and review of past actions. The checks and balances have as a corollary the pluralistic structure which tends to ensure support for alternative approaches to a problem. When an error cannot be hidden from a responsible and conscientious supervisory body, it is less likely to be repeated, widespread insight is generated concerning what should not be done, and bad administrators are contained in good time. It could be that in England funds at present spent for research and other academic purposes are sometimes wasted and scientists discouraged through poor judgment, and the consequences are not subject to adequate scrutiny. If this is so and there are less than good administrative beliefs and practices that will proceed unchecked, a large increase in research expenditures within the present framework may not provide the fullest of returns.

11. Conclusion

11.1 The C.C.L. Experiment

This monograph was written to report in a convenient form the work on computer-aided typesetting that was done in the C.C.L. at M.I.T. The author started the C.C.L. typesetting project to explore the possibilities of programming a computer to produce operating tapes for typesetting machines. An initial experiment led on to further studies, and the project expanded as mentioned in the Preface. The emphasis throughout the work was on an open-minded exploration of the unknown, as a research project which had been formulated and started and was being pursued in the context of a university, whose members are free to work towards the scientific goals and to investigate the scientific matters that appeal to their personal imagination, interest, and curiosity. The project was conducted for a period of three years, in this spirit of university research. Results were reported at appropriate junctions in the scientific literature,[10, 14, 17, 20, 34–36] and this reporting has now been consolidated with some further relevant material in the present monograph.

When the author started to work on computer-aided typesetting, he knew of no one else who was considering the field. By the time that the first public mention was made of his project, he had learned of the earlier Russian translation work which was using Linofilm, and in subsequent months he learned of the interest elsewhere in computer editing of Flexowriter tapes, in programmed hyphenation, and of the then proposed MEDLARS project. Millions of dollars will be spent annually in coming years on computer-aided typesetting, as a result of a variety of projects that are going on in the United States. Some specialized systems are in operation, as described in Chapter 8, and others of greater generality can be expected to follow in due course. Strong efforts are being made from various sources to direct the attention of the printing and publishing industry in several countries toward using computers in their typesetting work. The author hopes that computer-aided typesetting will bring widespread benefits in due course. He has assumed that the industry would progressively accept and make good use of developments which were duly reported and would invest its own efforts in a relatively spontaneous manner, if and when it became apparent that such actions were appropriate. The work that is discussed in Chapters 1 to 7 of this monograph was only experimental, but reporting it may encourage further experiments and help some other workers in their efforts to design, write, and operate more effective systems. Various systems and services are being offered to industry by commercial organizations already. It is to be hoped that more will follow and that they will be successful. Further comments on this score are not appropriate here, but it does not seem that the present reporting of the C.C.L. work, as an experimental study, is made redundant by these other efforts. It seems likely, in fact, that the scope and facility for further experiments and the need for their reporting in conventional scientific format will be increased, rather than diminished, by the introduction of computer-aided typesetting systems in industry and government.

11.2 Future Research

The work on computer-aided typesetting that has proceeded in the C.C.L. and elsewhere suggests several topics for further investigation:

1. The development of conventions for specifying typographic control that are more powerful and more convenient than those available at present, and the development of corresponding computer programs.
2. The development of algorithms and of corresponding programs to deal with hyphenation, rivers, and similar typographic details more effectively than at present.
3. The further development of input conventions and programs to deal with built-up formulas, page composition, and other "two-dimensional" typographic details.
4. The development of further editing systems.
5. The detailed market analysis needed to determine where the

cost reductions that could be provided by CATP's could be most helpful in stimulating increased production.

6. The study of the learning and relearning habits of keyboard operators and the optimum design of keyboard equipment.

Aspects of input conventions, program design and operation, editing commands, and keyboard practice that could be improved have been discussed in several chapters of this book.

The further development of CATP's can be regarded as an extension of some of the present activities of the printing-publishing industry. It can also be viewed in another perspective as part of the general advance of information science. CATP's will flourish spontaneously where the rest of information science flourishes and provides the incentives and the physical and intellectual wherewithal to develop CATP's. On a larger canvas, information science will flourish where the rest of science and scholarship are advancing, and it will contribute reciprocally to their further progress.

This need for appropriate technological, intellectual, and administrative climates to make new techniques usable and to provide enough demand for them is sometimes missed in exhortations that strive to introduce to one country processes that are finding use elsewhere. Computer-aided typesetting may be a case in point, and it could be of interest accordingly to recapitulate a few of the ways in which CATP's are made possible and necessitated by developments in the United States.

11.3 The Computer Culture and CATP's

First and foremost, the United States has a highly developed computer culture. Considerable numbers of computers are installed in industrial, governmental, and academic institutions and are used for a wide diversity of applications. The general standard of industrial development ensures excellent design sophistication, components, manufacturing techniques, quality control, reliability, and maintenance, in part as a result of the application of computers to several of these selfsame matters. Powerful compilers and other programming systems are available. These factors lead to widespread and effective usage of computers. The cumulative experience of users that is exchanged in organizations such as SHARE is mutually beneficial and feeds back to influence further hardware and software services and design. When new computer applications are envisaged, ready access can be gained to computing equipment and to personnel with the technical knowledge needed to use computers, and with the insight and experience needed to make sound administrative and procedural decisions. The importance of this last factor must not be underestimated. Computers have now been introduced to so many industries and activities that a few patterns of sound development have become apparent and widely known, and can be used when a new field of application is started. Conversely, the sorts of things that go wrong in such enterprises are well known and can be avoided. If a company wishes to explore the application of a computer to a type of problem for which computers have never been used before either there or anywhere else, the company may well have installed a computer already for more conventional applications, or it may be able to do so using knowledge of the existence of these conventional applications, and of the ways of handling them on a computer, gained by other companies in the same and allied industries.

Computer usage and program design are activities of many specializations that are highly developed in the United States. The fact that powerful compilers are as necessary as powerful hardware has been a commonplace in popular accounts of computing in the United States for many years. There are, moreover, numerous other programming systems in use in the United States, the development of which required efforts that were comparable with those expended on compilers. Time and time again research work using computers goes forward because a scientist concerned with one field of endeavor can use an elaborate program written by people in a neighboring field to link up with the programs that he has written.

The prevalence of computers and computational experience and insight makes a new use of computers seem quite natural in the United States where elsewhere the introduction of a computer for the same sort of application, but in "cultural isolation," would be dramatic and perhaps hazardous. Typesetting applications are no exception in this respect. Again, the general level of technical and financial activity and within this the prevalence of computer utilization in the United States is leading to vast quantities of computer-generated material to be set in type. While keyboard work that is faster and not limited by present union-formulated prerequisites attracts considerable attention, CATP's that have this as their sole objective may well be carried by the momentum of CATP's which set material that the computer has produced or uses in other ways.

It is very likely that one way in which CATP's will swell into widespread use is as an incremental activity of typesetting shops which have installed tape-operated equipment for conventional use, and an activity of computer installations that were set up for nontypesetting computer applications. As typesetting programs became available, more and more existing computer installations will be led to convert results to operating tapes for running in local typesetting shops that have appropriate tape-operated equipment. This will lead naturally to the organizations in which the computers are installed considering further CATP's that are relevant to their activities, and to typesetting shops increasing their capacity to deal with computer-generated tapes. The program maintenance and supplementary programming that are mentioned in Section 9.4 will be carried as incremental efforts by the computer installation staff employed already on their companies activities, with the resulting insight into how their environment can be benefited by further CATP's. This will proceed apace in a spontaneous and widespread manner which is geared to a general trend in computer usage, and to which exhortations and directives to the typesetting industry in a computationally moribund society are utterly irrelevant.

11.4 Associated Activities

There are several further foci of technical activity besides computing that are cogent to the development of CATP's in the United States, particularly those involving high-speed typesetting devices. Communications technology is making rapid progress, and communications capacity is expanding rapidly to meet the increasing needs of voice and video transmission and data telemetry, and of digital information transfer to and from computer systems. The volume of digital data transmission over communications systems in the United States is likely to exceed the volume of voice transmission within a few years. These developments, in addition to swelling the bulk of computer results that may need to be typeset, will also be conducive to the increased use (and cost sharing with CATP's) of decentralized high-speed typesetting equipment to disseminate material which is to be read.

The expanding and evolving requirements of a diversity of industries encourage advances in the chemistry and physics of industrial materials, photography, and other copying processes, and advances in the component and personnel availability for optical and electronic work, which all have an obvious relevance to the further development of CATP's.

There are several trends now evident in computer usage that require, and whose further study will be stimulated by, developments in CATP's. These trends are discussed here under a few main headings between which there is a considerable overlap.

11.5 CATP's and Information-Generating Procedures

Computers are used extensively to generate "new" information. This may be in one of several forms:

1. Numerical, for example, insurance tables when new rates are introduced.
2. Symbolic, for example, mathematical and chemical formulas.
3. Verbal nonsentential, that is, material that consists entirely or partly of words which are not organized in sentences, for example, job assignments, routing instructions, and other schedules.
4. Sentential, for example, servicing instructions for mechanically designed equipment.

These uses 2 to 4 correspond to an increased understanding of nonnumeric processes, and with hardware and software facilities to deal with these. Uses 3 and 4 relate to the increasing employment of computers to plan operations and to design equipment. These uses are necessitated by (*a*) the numbers and dispersal of personnel and diversity of processes and products that are involved in activities which must be integrated, (*b*) the complexity of many modern equipment systems, (*c*) the speed with which plans and designs must be made, sometimes as the reaction to sudden external developments, and (*d*) variations in product specifications to meet the requirements of individual users of products which are being manufactured.

11.6 CATP's and the Transformation of Information

Mechanical language translation has not developed as rapidly as its exponents had hoped would happen. There are, however, several aspects of text processing which are analogous to, or part of, the full language translation problem, but which are rather less difficult. These may achieve extensive utility before full natural language translation by machine has been achieved, and will require typeset output facilities to be most effective. Translation between different levels of stylization of the same natural language could be used, say, to produce versions of a manual that were appropriate to different prerequisite vocabularies and knowledge. Translation from a stylized version of an author's language to simple forms of numerous other languages and dialects could be of use in preparing manuals for rapidly developing countries. Related processes are relevant to the production of reports appropriate to specialized interests that are mentioned in the section which follows. The full mechanical language translation problem may be solved adequately some day, and its solution would then open the floodgates of necessity for further CATP's.

11.7 CATP's and Information Retrieval

Computers are used increasingly to record information in reference systems from which items relevant to user inquiries can be retrieved. Computers are needed in this work because information is being generated and collected by professional personnel throughout the world at a vast and increasing rate, partly as a result of developments in scientific observing equipment and of computers themselves. Cumulative records of this information must be kept current and often must be pooled from numerous sources and made available to many inquirers rapidly.

Reference systems may be for manual visual use (e.g., frequently revised books and card catalogs) or for mechanical use, recorded on media which form part of, or can be attached to, a computer which is then used in the retrieval as well as in the recording process. In the former case, printing is necessary to update the manual visual system. In the latter case, printing is necessary if text matter is retrieved by the computer, but not to the same extent if only citations are retrieved.

Numerous forms of index can be involved in this type of work, with index terms selected by author or by other persons or by machine. The development of systems of index terms for uniform use throughout individual fields of endeavor is itself a matter that has been stimulated by prospects of computer retrieval and now commands considerable attention. The production and maintenance of conventional subject, author, title, and chronological indexes and world and union lists, and of citation, keyword in context, and permuted title indexes benefit considerably from the use of the computer in scanning, sorting, merging, and file-inverting operations. Depth and correlative indexing, "collecteana," and other referencing devices and ways of organizing stored information and representing it concisely are being explored further as the extent to which computers facilitate their use increases.

The methods that are used to retrieve stored information are related to the way material is organized in storage. Work is progressing with the use of concept coordination and other index-dependent and index-independent approaches to selecting items that are relevant to a user's interest. The production of specialized bibliographies and indexes and compendia of selected text items is a current objective of several U.S. government agencies.

A differentiation is evolving between the methods that are used to deal with the "current awareness" problem and with retrospective literature searching. The so-called SDI (selective dissemination of information) approach is an example of a software development pertaining to current awareness. The hardware and software developments in message routing disclose further vistas of relentlessly sustained awareness. Remote consoles, time sharing, intercomputer communication, and so forth will all extend the possible location of questioners and lessen the time of answering, and make the escape from current awareness more and more difficult. Some of this will undoubtedly be of use to the decision makers and hopefully to those affected by the decisions. Moreover, despite the high stage of evolution of computers that are organized around arithmetic processing units, time may see the development of further classes of information processing machine not arithmetically organized, which would be of far greater power than present equipment for nonnumeric processes such as language translation, and for processing continuum information such as handwriting and speech.

11.8 The Future

Where these various matters will all lead to, with CATP's merrily producing printed pages *ad infinitum*, remains to be seen. Of course, society need not be saturated with the rate of production of printed matter while anyone has any time left to read, and even then, there can still be Gatsby's to impress latter-day Owl's-Eyes with books in unused libraries. It was hoped at one time in the C.C.L. that the development of some editing programs would help solve the documentation problems of the laboratory, but the need to document these programs increased the problems by an order of magnitude. It seemed that the motif of Greek tragedy was to be re-enacted, with the efforts that were being taken to avoid a predictable doom leading inexorably toward it. Now there is the real prospect of computer-aided typesetting expediting the extension of computer-aided typesetting, and the theme of *L'Apprenti Sorcier* returns with books in place of water. CATP's may well augment the flood of printed pages that are produced in the cause of academic tenure, managerial competition, ego satisfaction, and the fear of making decisions that seeks protection in anthropomorphizations of the digital computer. Nonsense will gain the authority of the well-typeset page in much greater quantities than now. But the good that can come from the progress of CATP's within the advancing front of science and scholarship will not be lessened. And as long as there are contributions that can be made to CATP's and like matters by university scientists, it is to be hoped that universities will continue to make such contributions possible.

References

1. *Technical Manual for Flexowriter Programatic Automatic Writing Machine*, Form SP–8673–Roch., Friden Corporation, Rochester, N.Y., 1962.
2. *Reference Manual: 46 and 47 Tape-to-Card Punches*, Form A24–1064, IBM Corporation, 1956.
3. *Reference Manual: 709 Data Processing System*, Form A22–6536, IBM Corporation, 1960.
4. *Reference Manual: 709/7090 FORTRAN Operation*, Form C28–6066, IBM Corporation, 1963.
5. *Reference Manual: IBM 709/7090. Programming Systems: FORTRAN Assembly Program (FAP)*, Form C28–6235, IBM Corporation, 1963.
6. *Reference Manual: 709/7090 FORTRAN Monitor*, Form C28–6065, IBM Corp., 1960. (In later editions this is combined with form C28–6066.)
7. *Reference Manual: 1012 Tape Punch*, Form A26–5776, IBM Corporation, 1964.
8. *Principles of Operation of the Photon S–560 System*, Photon, Inc., Wilmington, Mass., 1963.
9. "Computer Editing of Verbal Texts. Part I. The ES1 System," M. P. Barnett and K. L. Kelley, *American Documentation*, **14,** 99 (1963).
10. "Surface Harmonic Expansions of Products of Cartesian Coordinates," A. B. Otis, Jr., and M. P. Barnett, *Mathematics of Computation*, **18,** 635 (1964).
11. *Reference Manual: 24 and 26 Card Punches*, Form A24–0520, IBM Corporation, 1963.
12. "The Evaluation of Molecular Integrals by the Zeta Function Method," M. P. Barnett, *Methods in Computational Physics*, B. Alder, *et al.*, eds., Vol. 2, *Quantum Mechanics*, Academic Press, New York, 1963, p. 105.

13. "Mechanization of Tedious Algebra—the e Coefficients of Theoretical Chemistry," H. D. Wactlar and M. P. Barnett, *Communications of the Assoc. for Computing Machinery,* **7,** 704 (1964).
14. *Computer Generated Formulae for the e Coefficients of Theoretical Chemistry*, H. D. Wactlar and M. P. Barnett, C.C.L. Technical Note 31, M.I.T., 1963.
15. "MW9BCD BCD Manipulative Routines," M. J. Bailey, P. B. Burleson, E. J. D. Carter, and K. L. Kelley, SHARE Distribution No. 1371 (Nov. 20, 1962).
16. "Mechanization of Tedious Algebra: The Newcomb Operators of Planetary Theory," J. M. Gerard, I. G. Izsak, and M. P. Barnett, *Communications of the Assoc. for Computing Machinery*, **8,** 27 (1965).
17. *Construction of Newcomb Operators on a Digital Computer*, I. G. Izsak, J. M. Gerard, R. Efimba, and M. P. Barnett, Special Report No. 140, Smithsonian Institution Astrophysical Observatory, Cambridge, Mass., 1964.
18. "Synthesis of Two Dimensional Representations from Linearized Descriptions," J. M. Gerard, unpublished Ph.D. dissertation, University of London, 1965.
19. "Syntactic Analysis by Digital Computer," M. P. Barnett and R. P. Futrelle, *Communications of the Assoc. for Computing Machinery*, **5,** 515 (1962).
20. "Computer Generation of Photocomposing Control Tapes. Part I. Preparation of Flexowriter Source Material," M. P. Barnett, K. L. Kelley, and M. J. Bailey, *American Documentation*, **13,** 58 (1962).
21. *Graphic Composing Techniques*, RADC No. IDR 61–310, Final Report, Contract AF 30 (602)–2527, Thomas J. Watson Research Center, IBM Corporation, Yorktown Heights, N.Y., Feb. 1, 1962.
22. *The Linofilm System*, Form LF 1.04–H–XX–35X, Mergenthaler Linotype Company, New York, 1964.
23. *Phototypesetting of Computer Output: An Example Using Tabular Data*, W. R. Bozman, Technical Note 170, National Bureau of Standards, Washington, D.C., June 25, 1963.
24. *Experimental Transition Probabilities for Spectral Lines of Seventy Elements*, C. H. Corliss and W. R. Bozman, Monograph 53, National Bureau of Standards, Washington, D.C., July 20, 1962.
25. "Computer Typesetting—An Evaluation of the Problem," C. J. Duncan, J. Eve, L. Molyneux, E. S. Page, and M. G. Robson, *Printing Technology (London)*, **7,** 133 (1963).
26. "The Computer and Newsprint," E. Yasaki, *Datamation*, **9,** 27 (March 1963).
27. "Monitor Controlled Composition of Bookwork," R. Boyd, *Computer Typesetting Conference Proceedings*, Institute of Printing, London, 1964, p. 25.
28. *The MEDLARS Story at the National Library of Medicine*, U.S. Department of Health, Education and Welfare, Washington, D.C., 1963.

29. "The Linotron System," V. M. Corrado, *Computer Typesetting Conference Proceedings*, Institute of Printing, London, 1964, p. 212.
30. "Physical Principles of Displays-Classification," Harvey G. Talmadge, Jr., *Electronic Information Display Systems*, J. H. Howard, ed., Spartan Books, Washington, D.C., 1963, p. 71.
31. *Programmers' Reference Manual S-C 4020 High-Speed Microfilm Recorder*, E. A. Bates, form 9,500,000, General Dynamics Electronics, San Diego, April, 1964.
32. "Computer-Made Perspective Movies as a Scientific and Communication Tool," E. E. Zajac, *Communications of the Assoc. for Computing Machinery*, **7,** 169 (1964).
33. *Student Political Attitudes*, M. Spiers, Bradford Institute of Technology, Bradford, England, Aug. 1964.
34. "Computer Generation of Photocomposing Control Tapes Part II: The PC6 System," M. P. Barnett, D. J. Moss, and D. A. Luce, *American Documentation* **15,** 115 (1964).
35. "Operating Experience with a Photon 560 Installation," D. A. Luce and M. P. Barnett, *Automation and Scientific Communication*, Proceedings of the ADI Annual Meeting, Washington, D.C., 1963, p. 293.
36. "Computer Controlled Printing," M. P. Barnett, D. J. Moss, D. A. Luce, and K. L. Kelley, *1963 Spring Joint Computer Conference* (Proceedings), AFIPS, Washington, D.C., 1963, p. 263.

Index

a and b coefficients, 7, 92
AB code, 44–45, 47
AL code, 44–45, 47
Annotation of manuscripts, 25, 73–75, 194–195, 205; *see also* Typographic control codes
AP code, 52
Assembly programs, 143, 220
Automated set cards, 13

Backtracking procedures, 156–157
Basic machine unit, 18
Batch processing, 142
BCDPRINT,
  applications, 100–114
  codes, 101–102, 107, 113
  preprocessor, 146
BD codes, 39–42, 45, 47
Benefits of computer typesetting, 193
Bibliography production, 120–125; *see also* Geological Survey, *Index Medicus*
Body lead, 17
Built-up formulas; *see* Mathematical formulas
Burroughs B-305 computer, 187

C sheet, 18
Capital investment, 194
Capitalizing, rules of, 126, 128, 148–151
Cash register, 210, 211
Catalog of Lunar Craters, 7, 69, 86
Catalogs, 3, 190, 233; *see also Catalog of Lunar Craters*, *Current Serials and Journals* catalog, "Picture Book," *Student Directory*
Centering, 50
CH code, 52–54
Character disc; *see* Photon 560 disc
Charactron, 186
Code insertion, 191
Code translation, 191
Columns; *see* Tables
Column headings, 88, 95
Communications systems, 190, 191, 231
Compilers, 143–145, 229; *see also* FORTRAN
Composite codes; *see* Macros
Computers; *see* Burroughs, IBM, NCR, On-line use, RCA, Remote operation, Time sharing
Console; *see* Control unit
Control codes; *see* Typographic control codes
Control unit, 82–83
Convenience of control codes, 75–76, 120–125
Cooperative Computing Laboratory, 8
Corrections; *see* Errors
Costs,
  equipment, 191, 194–196, 198–199
  personnel, 194–196
  programming, 105, 196–198
  running costs, 194

CP code, 52
Current awareness, 234
Current Serials and Journals catalog, 125–128, 148–151
Cyclic format, 120–125, 162–163

D sheet, 18
Delay modifiers, 161–162
Diagnostics, 221
Directories; *see* Catalogs
Disc; *see* Disc store, Photon 560 disc
Disc file; *see* Disc store
Disc level; *see* DL code, Photon 560 disc
Disc store, 142
DL code, 27, 30–33, 37–39
Documentation; *see* Mechanized documentation, Program documentation
Double spacing; *see* VL code
DP code, 52

e coefficients, 7, 103–109
Economic benefits, 193–207; *see also* Benefits of computer typesetting, Costs
Editing programs, 65–66, 77–79, 129–135, 146, 151–153, 154–155, 204, 227, 229
Electron-tube devices, 185–187
EN code, 30–33
Errors,
  bypassing, 81–82, 221
  causes, 79, 114, 119–120, 174, 199, 200
  correction, 64–66, 77, 194, 221; *see also* Editing programs
  frequency, 194
  location, 79–81, 194
ES1 system, 151–153
Escapements, 18
Expand code, 159
External coding, 3, 162–163; *see also* TABPRINT
Extraction processes, 190

FC code, 50, 61
Film magazine; *see* Photon 560 film magazine
Film preparation, 20–22, 67
Filmsetting machines; *see* GRACE, Linofilm, Linotron, Photon, ZIP
FL code, 50, 61
Flash lamp; *see* Photon 560 flash lamp
Flexibility; *see* Programming tactics
Flexowriter, 2, 8–9, 25, 103, 113, 118, 166, 167, 181, 212–213
Font, 26, 37–39
Font specification, 30
Footnotes, 159
FORMAT statements, 96
Forms, 139
FORTRAN, 11, 96, 111, 164, 220
FR code, 50, 61
Fragmentation; *see* Programming tactics

Geological Survey, 187
Government Printing Office, 186–187
GRACE, 184–185

HI code, 55
Hollerith codes, 85
HS code, 55
Hyphenation, 165, 174, 176–180, 191, 204, 215, 217

IBM editing programs, 188
IBM machines,
  026 card punch, 103
  047 tape-to-card converter, 11–12, 103, 118, 130
  407 tabulator, 126, 127
  704 computer, 167
  709 computer, 2, 8, 141
  722 unit, 12
  870 document writer, 130
  1012 tape punch, 11
  1401 computer, 11
  1403 printer, 188
  1620 computer, 179
  7090 computer, 11
IBM Russian translation project, 166–172, 227
Image reversal, 22
IN code, 30
Indention, 47–50, 158
Index Medicus, 166, 180–185, 201, 227
Indexes, 233–234
Information generation, 232
Information retrieval, 233
Information science, 229–235
Information transformation, 232
Input, 3, 9, 64, 87, 115–119, 131–132, 136–137, 141, 189, 190
Insurance tables, 232
Internal codes; *see* Interspersed codes
Interpretive codes, 157
Interspersed codes, 3, 139; *see also* TYPRINT, BCDPRINT
Intertype machines; *see* Linecasting machines

JU code, 50
Justification, 31, 50, 55

Keyboard operations, 76, 191, 201–206

Keyboard operators, 214–218, 229
Keyboard perforators, 189; *see also* Flexowriter, TTS perforator
KP code, 52

Language translation, 165, 166–172, 232
LD codes, 39–42, 45, 47
Leaders, 159–160
Leading; *see* Body lead, Vertical spacing
Left adjustment, 50
Lens; *see* Photon 560 lens
Letter spacing, 158
Library catalogs; *see Current Serials and Journals* catalog, Union lists
Ligatures, 146
Line breaks, 31, 35–37
Linecasting machines, 176–179, 210, 211
Line length, 31, 35–36, 50
Line printers, 85
Linofilm, 95, 166–174
Linosec, 179–180, 204
Linotron, 186–187
Linotype; *see* Linecasting machines
LM code, 56
LN code, 35–37, 50
LO code, 55
LP code, 52
LS code, 21–31, 37–39

Machine Composition Company, 8
Machine language, 143
MA code, 47–50
Macros, 63–64
Magnetic tape, 85, 141
Man-machine interaction, 179–180
Margins, 47–50, 56, 160
Market analysis, 228
Mathematical formulas, 104, 106–107, 135–139, 188, 192, 228
Mathematical symbols; *see* Special characters
Mathematical tables; *see* Tables
Mechanical translation; *see* Language translation
Mechanized algebra, 103–114
Mechanized documentation, 232–234
Mechanized indexing, 233–234
Mechanized syntactic analysis; *see* Syntactic analysis
MEDLARS; *see Index Medicus*
Merging, 190
Modularity; *see* Programming tactics
Molecular integrals, 6, 68, 88, 89
Monitor programs, 141–142
Monotype machine, 174, 176
Multiple spacing; *see* VL code

National accounting machine, 210, 211
National Bureau of Standards table setting, 172–174
National Library of Medicine; *see Index Medicus*
N.B.S. table setting, 172–174
N.C.R. 315 computer, 179
Newcastle project, 174–176
Newcomb operators, 7, 109–114
Newspaper applications, 179
NJ code, 50
NL code, 36
N.L.M; *see Index Medicus*
NM code, 47–50
Nonjustified output, 50
Nonnumeric computer applications, 103; *see also* Mechanized algebra, Syntactic analysis, Text processing
Nonset mode, 160–161
NP code, 33–35

Object code editing; *see* "Picture Book"
On-line input, 192
Operators, 194–196, 209; *see also* Keyboard operators
Organization processes, 190, 211–212
Output, 142

Page breaks, 20, 52, 97–98
Page length, 31, 50, 52
Page make-up, 192, 229
Page numbers, 52, 160
Paragraph breaks, 33–35, 158
Parameters, 157, 222
PC6 programs, 1, 3–7, 25–135, 145–147, 153, 156–157
PD code, 52
Perforators; *see* Keyboard perforators
Personnel requirements, 194–196, 208–209
Personnel training, 209–218
PG code, 52
Photon 200 phototypesetting machine, 8, 13
Photon 513 phototypesetting machine, 13
Photon 540 phototypesetting machine, 13
Photon 560 phototypesetting machine, 2, 8, 12–24,
  codes, 22
  disc, 13–15
  film, 19–22
  film magazine, 19–20
  flash lamp, 13–15
  lens, 15–17
  lens turret, 15–16

Photon (*continued*)
  pi-mat, 22
  prism, 17–19
  tape, 22–24
Photon ZIP, 206; *see also* GRACE
Pianola, 210, 211
"Picture Book," 100, 128–135
Pi-mat; *see* Photon 560 pi-mat
Planetary theory; *see* Newcomb operators
PL code, 52
PN code, 52
Point, 3, 18
Point size, 18
Prefix codes, 139
Preprocessors, 4–5, 142, 145–147
Probation pamphlet, 7, 72
Procedural language, 163–164
Program documentation, 197, 221
Program maintenance, 197
Programming tactics, 140, 143–145, 156–157, 128–222; *see also* Compilers
Proofreading; *see* Error causes, Error correction
Proprietary programs, 198
Punched-card machines, 84–85, 103, 125, 190, 191, 210, 212–213

Quality, improvement of, 193
Quantum chemistry; *see* *e* coefficients, Molecular integrals

RCA 301 computer, 179
Rapidly developing countries, 217
Reflecting prism; *see* Photon 560 prism
Relative width, 18
Remote operation of computer, 193
Repetition, 190
Research,
  academic, 222–226
  administration of, 222–226
  applications; *see* Scientific applications
  future research, 228–229
Retraining; *see* Personnel training
Revision, 190
Right adjustment, 50
Right-reading film, 21
RM code, 56
Run failures, 79
Run processing, 66–67
Runarounds, 161
Running heads, 159
Running speeds, 194
Russian translation; *see* IBM Russian translation
ρ-code, 4–5, 145–147
ρ-ω translator, 4–5, 75, 153, 156–157

S codes, 56
SC 4020 machine, 186
Scanning, 143; *see also* Syntactic analysis
Scientific applications; *see* Geological survey, *Index Medicus,* N.B.S., Newcomb operators, Quantum chemistry, Scientific instruments
Scientific instruments, 190, 191
Sections; *see* LM, RM, XS codes
Set size, 19, 159
SHADOW, 151
Social attitudes, 216–218
Sorting, 190
Source programs, 143
Spacing,
  horizontal, 36, 54–55
  interword, 55, 158; *see also* Justification
  vertical, 20, 39–47
Special characters, 51, 52–54
Special-purpose computers, 207; *see also* Linosec
Standing files, 190
Standing type, 212
Storage, 142
String transformations, 142
Student Directory, 128
Style; *see* Font
Stylized instructions, 151–153
Subprograms, 144–145, 219
Subscripts, 16, 69, 111, 187
Superscripts, 16, 69, 108, 111, 187
Supplementary programming, 197
Syllabus, 209–214
Syntactic analysis, 147–153, 154–155

Table headings, 87, 95
Tables, 55–63, 84–99, 172–174
TABPRINT,
  applications, 4, 6–7, 89–92
  codes, 86–88, 90, 93–94
  operating experience, 94–99
  preprocessor, 166
Tape-to-card converter, 9–11
Technological evolution, 229–231
Text processing, 142, 143; *see also* Language translation, Syntactic analysis
Theoretical chemistry; *see* Quantum chemistry
Time factors, 195, 199–200
Time savings, 193
Time sharing, 193, 199
TTS perforator, 210, 211, 215
Type alignment, 16
Type size, 16, 28–29, 37–39

Type style; *see* font
Typographic control codes; *see* BCD-PRINT, IBM Russian translation, TABPRINT, TYPRINT
Typographic functions, 5, 157–162
TYPRINT,
  applications, 3, 6–7, 68–72
  codes, 25–64
  operating experience, 73–83
  preprocessor, 145–146

Union List of Communist Chinese Serials, 7, 70, 114–125, 147
Union lists, 114

Vertical alignment, 50, 55–63, 87, 93–94
VL codes, 42, 46–47
VP codes, 43, 46

Wages, 194, 199
Wrong-reading film, 21

XS codes, 56–64

Zero leading, 20
ZIP, 206, *see also* GRACE